UNIVERSIDADE METODISTA DE PIRACICABA

FACULDADE DE CIÊNCIAS EXATAS E DA NATUREZA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

IMPLEMENTAÇÃO DE UMA POLÍTICA DE COLABORAÇÃO PARA UM AMBIENTE DE CO-AUTORIA DE DIAGRAMA DE CLASSES

CARLOS HENRIQUE RODRIGUES SARRO

ORIENTADOR: PROF. DR. LUIZ EDUARDO GALVÃO MARTINS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, da Faculdade de Ciências Exatas e da Natureza, da Universidade Metodista de Piracicaba – UNIMEP, como requisito para obtenção do Título de Mestre em Ciência da Computação.

PIRACICABA
2006

UNIVERSIDADE METODISTA DE PIRACICABA

FACULDADE DE CIÊNCIAS EXATAS E DA NATUREZA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

IMPLEMENTAÇÃO DE UMA POLÍTICA DE COLABORAÇÃO PARA UM AMBIENTE DE CO-AUTORIA DE DIAGRAMA DE CLASSES

CARLOS HENRIQUE RODRIGUES SARRO

ORIENTADOR: PROF. DR. LUIZ EDUARDO GALVÃO MARTINS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, da Faculdade de Ciências Exatas e da Natureza, da Universidade Metodista de Piracicaba – UNIMEP, como requisito para obtenção do Título de Mestre em Ciência da Computação.

PIRACICABA

2006

Sarro, Carlos Henrique Rodrigues

S247i Implementação de uma política de colaboração para um ambiente de co-autoria de diagrama de classes/Carlos Henrique Rodrigues Sarro - Piracicaba, SP:[s.n.], 2005.

Orientador : Luiz Eduardo Galvão Martins.

Dissertação (Mestrado)– Universidade Metodista de Piracicaba, Faculdade de Ciências Exatas e da Natureza, Programa de Pós-Graduação em Ciência da Computação.

1. Política de colaboração. 2. CSCW. 3. Groupware. 4. Sistemas distribuídos. 5. UML. 6. Diagrama de classes. I. Martins, Luiz Eduardo Galvão. II. Universidade Metodista de Piracicaba, Faculdade de Ciências Exatas e da Natureza, Programa de Pós-Graduação em Ciência da Computação.

IMPLEMENTAÇÃO DE UMA POLÍTICA DE COLABORAÇÃO PARA UM AMBIENTE DE CO-AUTORIA DE DIAGRAMA DE CLASSES

AUTOR: CARLOS HENRIQUE RODRIGUES SARRO

ORIENTADOR: PROF. DR. LUIZ EDUARDO GALVÃO MARTINS

Dissertação de Mestrado defendida e aprovada em 13 de Dezembro de 2005, pela Banca Examinadora constituída dos Professores:

Dr. José Luis Zem (UNIMEP)

Dr. Luiz Eduardo Galvão Martins (Orientador - UNIMEP)

Dr. Manuel de Jesus Mendes (UNISANTOS)

À

Minha esposa Luciane pelo apoio e compreensão e aos meus filhos David e Daniel pela paciência.

Aos

Meus pais Henrique e Stela pelo incentivo, carinho e amor.

AGRADECIMENTOS

A Deus primeiramente.

Ao professor Luiz Eduardo Galvão Martins a orientação, a compreensão e o Incentivo dispensado ao desenvolvimento deste trabalho.

À minha irmã Sandra e sobrinha Rafaela pela colaboração na revisão final deste trabalho.

A todos os funcionários da Instituição e aos professores Paulo E. Fabretti e Vitor Brandi Junior.

Aos pastores Nadyr Lautenschlager e João Francisco do Prado. À missionária Mara Bueno Consani.

“Ao que lhe disse Jesus: Se podes! - tudo é possível ao que crê.”

(S. Marcos 9:23)

RESUMO

Nas últimas décadas, muitos sistemas têm sido construídos para a área de CSCW, e, uma Política de Colaboração é importante no contexto de ambientes onde muitos usuários estão trabalhando ao mesmo tempo e em diferentes lugares. A política de colaboração proposta neste trabalho baseia-se em aspectos de papéis de usuários, tomada de decisão, mecanismos de comunicação e funcionalidades para assegurar a estabilidade e a flexibilidade do ambiente. Alguns desses aspectos são considerados fatores críticos de sucesso, pois devem ser aplicados para que os usuários tenham um bom desempenho (EDWARDS, 1996). Entretanto, esses mecanismos além de permitir um bom desempenho e confiabilidade na operação do ambiente, devem também eliminar o aspecto surpresa que é comum nos ambientes de CSCW, sem contudo perder o controle do ambiente. Neste trabalho desenvolveu-se uma ferramenta, em nível de protótipo, para validar uma política de colaboração de co-autoria de edição de diagrama de classes, e que aborda os mecanismos e propriedades dos sistemas.

PALAVRAS-CHAVE: Política de Colaboração, CSCW, *Groupware*, Sistemas Distribuídos, UML e Diagrama de Classes.

ABSTRACT

In the last decades many systems were built in the CSCW (Computer-Supported Collaborative Work) area and a Collaborative Policy is important in environment where many users are working at the same time in different places. The collaborative policy of this paper is based on aspects of users role, making decision, mechanisms of communication and functionality that assured stability and flexibility in the environment. Some of this features are critical success factors that must be applied to improve user work performance (EDWARDS, 1996). Nevertheless, these mechanisms beyond to allow a good performance and confiability in the work environment operation, must eliminating surprises, without losing control. In this work it was developed a tool, presently a prototype, called Orbital, which validates a co-authorship cooperation policy of editing class diagrams and broaches the system mechanisms and properties.

KEYWORDS: Collaborative Policy, CSCW, GroupWare, Distributed System, UML and Class Diagrams

ÍNDICE

LISTA DE ABREVIATURAS E SIGLAS

ACM	Association for Computing Machinery
CSCL	Computer Supported Collaborative Learning
CMC	Comunicação Mediada por Computador
CASE	*Computer-Aided Software Engineering*
CSCW	Computer Supported Colaborative Work
DFD	Diagrama de Fluxo de Dados
ER	Engenharia de Requisitos
GROVE	*GRoup Outline Viewing Editor*
IP	***I**nternet **P**roto-col*
j2sdk-1.4.2	*Java2 standard development Kit, versão 1.4.2*
MER	Modelo de Entidade e Relacionamento
PCs	*Personal Computer*
OA	*Office Automation*
OMT	*Object Modeling Technique*
OO	*Oriented-Object*)
OOA	*Object-Oriented Analysis*
OOD	*Object-Oriented Design*
OOSE	*Object-Oriented Software Engineering*
ISO	*Open Systems Interconnection*

RMI	Mecanismo de RPC para programação orientada a objetos. Permite que um objeto invocar um método que está em outro espaço de endereçamento.
RPC	*Remote Procedure Call*
SO	Sistema Operacional
SGBD	Sistema de Gerenciadores Banco de Dados
SASSE	*Synchronous Asynchronous Structured Shared Editor*
SEPIA	*Structured Elicitation and Planning of Ideas for Authoring*
SEE	*Software Engineering Environments;*
TCP	***T**ransmission **C**ontrol **P**rotocol*
UML	*Unified Modeling Language*
UNIMEP	Universidade Metodista de Piracicaba
WYSIWIS	*What you see is what I see*
WYSIWIMS	*What You See Is What I May See*

LISTA DE TABELAS

LISTA DE FIGURAS

1. INTRODUÇÃO

O CSCW (Computer Supported Colaborative Work) é uma área nova de pesquisa dentro do mundo da computação, tendo como objetivo para oferecer soluções e propostas para os trabalhos realizados em grupos, com apoio de computadores. Poucos trabalhos são executados sem a cooperação mútua, pois na maioria dos casos, algum tipo de colaboração é exigida para a realização de determinadas tarefas, seja em todo o trabalho ou em parte dele. A colaboração tem sido um requisito importante em muitos seguimentos e tipos de trabalhos, como escritórios, fábricas, ensino etc, que se utilizam da computação como suporte para a realização das suas atividades.

1.1. OBJETIVO DO TRABALHO

O objetivo deste trabalho é propor uma política de colaboração para um ambiente de co-autoria de diagramas de classes. A ferramenta de co-autoria deve auxiliar na modelagem de sistemas, baseada no diagrama de classes da UML (Unified Modeling Language). Num ambiente de CSCW os integrantes situar-se-ão geograficamente dispersos, porém, interagirão com a ferramenta, acessando operações em outros locais, remotamente, para a realização de um único trabalho, ou seja, um trabalho em grupo (colaborativo). Para isso, visualizarão o que os outros estarão fazendo, comunicando-se uns com os outros, em tempo real, a fim de resolver impasses, propor novos elementos na área comum de trabalho, opinar sobre novos elementos propostos por outros colaboradores. Através das tomadas de decisões, por exemplo, inserirão informações nos elementos publicados, resultando num modelo de classes inicialmente pretendido.

A política de colaboração proposta proporcionará estabilidade ao ambiente para que este não se torne caótico. Para isso, houve necessidade de se definir uma política de controle de acesso, chamada de papéis, que disponibilizasse mecanismos de controle sem perder a eficácia e o dinamismo

das interações entre os colaboradores. Tal qual no mundo real, os ambientes que suportam a colaboração precisam de regras e controles para um bom funcionamento.

1.2. JUSTIFICATIVA

Justifica-se a realização deste trabalho pela importância que tem para a proposta de desenvolvimento de uma ferramenta de co-autoria de diagramas de classes em ambientes colaborativos, multi-usuários (*Groupware*). Em termos de pesquisa, visa a busca de novos conhecimentos e técnicas de como aplicar os controles corretos em ambientes que são potencialmente caóticos.

Existem poucas ferramentas colaborativas disponíveis que se propõem a suportar trabalhos em grupo, principalmente entre as ferramentas *CASE*, onde há um freqüente questionamento por parte dos desenvolvedores de *software* de que tais ferramentas não suportam a colaboração, ficando para um único desenvolvedor a atividade de modelagem. A falta desse requisito também foi apontada com uma das causas de se abandonar o uso das ferramentas *CASE*, entre outros motivos.

A proposta de uma ferramenta com a mesma característica de uma CASE, que suporta o trabalho em grupo com total controle do ambiente, a fim de assegurar os requisitos de concorrência e controle de acesso, contribuirá significativamente para evolução das ferramentas CASE.

Como ponto de partida para a elaboração da proposta de uma política de colaboração, foi adotada uma ferramenta de edição de diagrama de classes da *UML,* denominada *Orbital v 1.0,* que foi utilizada como um estudo de caso.

Com a necessidade de transformar, evoluir, uma ferramenta mono-usuária, ou seja, que é operada por uma só pessoa, conforme ilustra a Figura 1, para uma ferramenta de co-autoria em um ambiente colaborativo, onde os usuários modeladores que estão remotamente distribuídos possam

construir um único diagrama, constituiu-se como um elemento motivador da proposta.

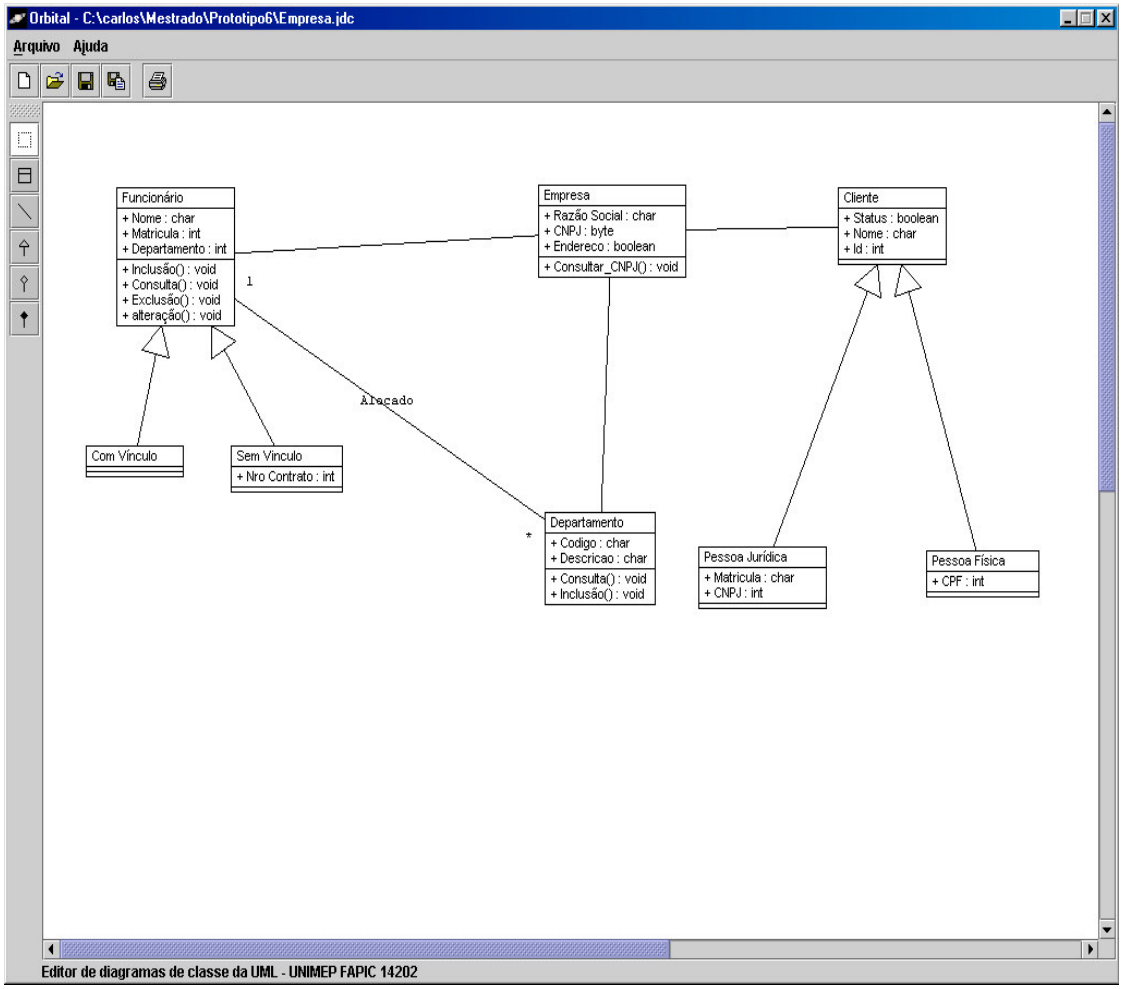


FIGURA 1 - ORBITAL (VERSÃO 1.0) DA FERRAMENTA DE EDIÇÃO DE DIAGRAMA DE CLASSE DA UML

Com a nova abordagem para a versão colaborativa da ferramenta, tornou-se necessário levantar os requisitos fundamentais (funcionais e não-funcionais) de uma política de colaboração. Para isso, o grupo de pesquisa em aplicações colaborativas, formada por professores e alunos da UNIMEP, foi envolvido para discutir e contribuir para a formação desses requisitos.

1.3. METODOLOGIA DE TRABALHO

A metodologia utilizada para a elaboração da proposta de uma política de colaboração dividiu-se em quatro etapas, conforme descrito a seguir.

a) Levantamento bibliográfico sobre:

- **CSCW**: explorando conceitos, classificações, tipos de ferramentas e trabalhos co-relatos que aplicam os conceitos de *Groupware;*
- **políticas de colaboração**: definições, objetivos, requisitos, modelos existentes e implementados;
- **UML** – *Unified Modeling Language* e seus diagramas com especial enfoque para o diagrama de classes.

b) Elaboração da proposta:

Utilizou-se de algumas atividades da Engenharia de Requisitos (ER) para levantamento dos requisitos fundamentais na fase de elicitação[1]. Para a modelagem dos requisitos levantados, foi utilizado o diagrama de casos de uso da *UML.*

c) Desenvolvimento da proposta.

Estudo da linguagem JAVA e de RMI, para o desenvolvimento do protótipo, possibilitando assim a validação da proposta e possibilitando o reaproveitamento das funcionalidades já implementadas na versão mono-usuária da ferramenta, que também estão escritas em JAVA, para a versão colaborativa da ferramenta.

[1] Vem do termo *elicitation* (inglês) que significa investigar algo que está obscuro num primeiro momento.

d) Validação da proposta.

Várias simulações foram feitas sobre o protótipo da ferramenta buscando observar os seguintes aspectos:

- configuração e desempenho do servidor, pois exerce um papel de suporte do ambiente colaborativo;
- acesso às funcionalidades apresentadas nos modelos de caso de uso, *menus* da ferramenta, checando se estavam de acordo com o proposto;
- configuração/customização do ambiente para a especificidade de cada projeto, como criação de papéis de usuários, formação do grupo de trabalho (modeladores), atribuição de direitos de acesso aos projetos e política de tomada de decisão;
- edição do diagrama, procurando observar o comportamento dos modeladores quando manipulam a área de trabalho local, a área de trabalho comum (pública) e a área de trabalho de outros desenvolvedores;
- controle de acesso e alocação dos elementos do diagrama de classes;
- monitoramento do ambiente para os elementos recém-publicados, convocação e apuração quanto à tomada de decisão;
- controle das regras de tomada de decisão.

Observou-se que a prototipação rápida, que foi utilizada na fase de elicitação e validação dos requisitos, teve um papel importante na compreensão da proposta por parte dos envolvidos, o que possibilitou um grau de discussão e compreensão adequados. A metodologia da ER e a técnica da prototipação evolutiva utilizada para desenvolver e validar os requisitos, foram fatores positivos e decisivos para a conclusão deste trabalho.

1.4. CONTRIBUIÇÃO DO TRABALHO

Além do objetivo geral deste trabalho, que é contribuir para a área de *CSCW*, propondo e implementando uma política de colaboração, espera-se também:

a) preencher uma lacuna na área de ferramentas *CASE*, pois há uma escassez de ferramentas colaborativas para engenharia de *software*;

b) contribuir com a produtividade das equipes de modelagem de *software*, em especial nas primeiras fases do desenvolvimento;

c) contribuir para a melhoria da qualidade dos modelos de classes produzidos pelas equipes de desenvolvimento;

d) contribuir para a área de *CSCW* explorando conceitos de políticas de colaboração.

1.5. ORGANIZAÇÃO DA DISSERTAÇÃO

O restante da dissertação está organizada da seguinte forma: no segundo capítulo é apresentada a revisão bibliográfica sobre a CSCW. No terceiro capítulo é apresentada a revisão bibliográfica sobre e ferramentas CASE com ênfase na colaboração. No quarto capítulo tem-se a revisão bibliográfica sobre UML com destaque no diagrama de classes. No quinto capítulo é apresentada a revisão bibliográfica sobre Política de Colaboração. No sexto capítulo é apresentada a proposta de uma política de colaboração para um ambiente de co-autoria de diagrama de classes. No sétimo capítulo são apresentados: a implantação do protótipo, a validação dos requisitos propostos e os testes realizados com o protótipo em diversos cenários. No oitavo capítulo, tem-se a Conclusão do trabalho e propostas para trabalhos futuros.

2. CSCW

2.1. INTRODUÇÃO

A tradução mais comum encontrada para *CSCW*, nas diversas referências bibliográficas é Trabalho Cooperativo Suportado por Computador ou pode ser também Trabalho Cooperativo Apoiado por Computador (NITZKE, 1999).

Conforme apresentado por Borges (1995), o termo g*roupware* costuma ser usado quase como sinônimo de *CSCW*. Porém, alguns autores identificam uma tendência diferenciada no emprego desses termos. Enquanto *CSCW* é usado para designar a pesquisa na área do trabalho em grupo e como os computadores podem apoiá-lo, *groupware* tem sido usado para designar a tecnologia (*hardware* e/ou *software*) gerada pela pesquisa em *CSCW*. *Groupware* corresponde aos sistemas baseados em computador que dão suporte a grupos de pessoas engajadas em uma tarefa (ou objetivo) comum e que provêem interface a um ambiente compartilhado (BORGES; CAVALCANTI; CAMPOS, 1995).

Duas características importantes proporcionadas pela tecnologia de *CSCW* são: (1) Transparência de localização; (2) Consciência de trabalho cooperativo. Ao mesmo tempo que os usuários possuem acesso pleno aos serviços do ambiente de forma independente da sua localização geográfica, há a consciência de que o trabalho que está sendo desenvolvido é resultado de um esforço coletivo. A combinação da transparência e consciência proporcionadas pelo *CSCW* produz um ambiente avançado adaptável para aplicações específicas (BORGES; CAVALCANTI; CAMPOS, 1995).

2.2. FATOS HISTÓRICOS

Para a ciência da computação, o suporte ao trabalho cooperativo é algo bem recente. Embora computadores sejam agora ferramentas populares, a indústria de *software* tem geralmente explorado o suporte ao trabalho individual. Processadores de textos, editores gráficos e planilhas eletrônicas, por exemplo, são ferramentas que foram construídas visando o apoio ao trabalho individual e não coletivo. Áreas como interfaces homem-máquina têm se dedicado a explorar mais a interação de um só indivíduo com a máquina (NITZKE, 1999).

Computer Supported Cooperative Work surgiu como uma área de pesquisa em meados da década de 80. Segundo Farias (2002), a área de *CSCW* tem como foco a realização de um mesmo trabalho, ou atividade, por várias pessoas ou por um grupo de pessoas. Nessa época, dois pesquisadores, *Irene Greif* e *Paul Casham,* deram origem a essa sigla, *CSCW*, como sendo uma nova área de pesquisa (HOFTE, 1998).

Poucos anos depois, em 1986, já foram realizadas conferências, tanto nos Estados Unidos [[2]CSCW86; CSCW88; CSCW90, CSCW92; CSCW94; CSCW96], organizada pela ACM, quanto na Europa [[3]ECSCW89; ECSCW91; ECSCW93; ECSCW95; ECSCW97]. Outras conferências de periodicidade irregular, bem como diversas conferências que visam primordialmente outros temas, têm dedicado um crescente espaço à área de CSCW (HOFTE, 1998).

A partir da CSCW, novas áreas de pesquisa surgiram, tal como a CMC (Comunicação Mediada por Computador), englobando o universo computador/comunicação e a CSCL, Aprendizagem Colaborativa Apoiada por Computador (NITZKE, 1999).

---

[2] *Conference in Computed-Supported Collaborative Work of USA*
[3] *Europe Conferece in Computed-Supported Collaborative Work*

Foi dessa área de pesquisa que surgiram os produtos chamados de *Groupware*, que têm como característica principal, a realização de trabalhos em grupo, num mesmo ambiente computacional, com interfaces compartilhadas (FARIAS, 2002).

Groupware é uma tecnologia projetada para facilitar o trabalho em grupos e pode ser utilizada para comunicar, cooperar, coordenar, resolver problemas, competir ou negociar. Enquanto tecnologias tradicionais, como a do telefone, sejam qualificadas como *Groupware*, o termo é primordialmente utilizado para referir-se a uma classe específica de tecnologias apoiadas em modernas redes de computadores, tais como *e-mail*, *newsgroups*, vídeos ou *chats* (COMO, 2000).

Em meados dos anos 70, a crescente preocupação em aumentar a produtividade das organizações, onde a maior parte do trabalho é feita em grupo, deu origem a uma área de pesquisa chamada Automação de Escritório (OA). Os primeiros esforços nessa área buscavam integrar e transformar aplicações mono-usuário, como processadores de texto e planilhas eletrônicas, de forma a permitirem o acesso simultâneo de um grupo de usuários. Só mais tarde reconheceu-se a necessidade de realizarem-se estudos sobre o comportamento dos grupos ao desempenhar uma atividade. Tais estudos serviram como base para gerar sistemas de suporte mais apropriados. Assim, técnicos aliaram-se a profissionais de áreas humanas, como por exemplo, sociólogos, psicólogos, antropólogos e educadores, buscando o desenvolvimento de tecnologias mais adequadas para dar suporte ao trabalho cooperativo (NITZKE, 1999).

Na década de 1980, essa nova tecnologia ficou restrita às Universidades e aos centros de pesquisas e, somente em 1990 os primeiros sistemas de *Groupware* foram desenvolvidos e comercializados (FARIAS, 2002). Mas foi o advento da Internet e dos PCs (*Personal Computer)*, computadores portáteis (*Note Book),* que fez com que a CSCW se disseminasse e ganhasse espaço entre seus usuários (FARIAS, 2002).

CSCW refere-se tanto aos produtos (*Groupware*), como também a um movimento social promovido pela ciência da computação com o intuito de proporcionar um melhor suporte computacional às pessoas, primeiramente no âmbito profissional (KLING, 1994).

O sistema *Groupware* oferece vantagens significativas para grupos de usuários. Algumas das razões mais comuns porque as pessoas querem utilizar *Groupware* (COMO, 2000) são:

- facilitar a comunicação: mais rápida, mais clara, mais persuasiva;
- permitir comunicação onde não seria possível em outra situação;
- economizar tempo e custos na coordenação de trabalho em grupo;
- facilitar a solução de problemas em grupo.

2.3. DIFERENÇAS ENTRE TRABALHO COOPERATIVO E COLABORATIVO

Segundo Tijiboy e Maçada (2004), a cooperação parte do princípio que dois ou mais indivíduos trabalhando conjuntamente, podem chegar a uma situação de equilíbrio, onde as idéias possam ser trocadas e distribuídas entre os participantes do grupo, gerando assim, novas idéias e novos conhecimentos, frutos desse trabalho coletivo. É um método no qual o indivíduo deve ser motivado para ser cooperativo ou cooperador. Ressalte-se que, em momento algum os componentes devem ser intimidados ou alguma idéia lhes ser imposta. Toda a discussão deve fluir livre e espontaneamente, caso contrário, corre-se o risco de incorrer em coação social.

> "O conceito de cooperação é mais complexo que o de interação e de colaboração, pois, além de pressupor, ambos requerem relações de respeito mútuo e não hierárquicas entre os envolvidos, uma postura de tolerância e convivência com as diferenças e um processo de negociação constante. Percebemos que a diferença fundamental entre os conceitos de colaboração e cooperação reside no fato de que para haver colaboração o indivíduo deve interagir com o outro existindo ajuda mútua. Para existir cooperação deve haver interação,

> colaboração, mas também, objetivos comuns, atividades e ações conjuntas e coordenadas" (TIJIBOY; MAÇADA, 2004).

Como muitos autores tratam cooperação e colaboração como sinônimos, existindo, porém, outros distinguem esses dois conceitos. Discutir-se-ão, a seguir, essas diferenças e/ou semelhanças (TIJIBOY; MAÇADA, 2004).

Segundo Ferreira (apud Barros, 1994), Colaboração significa trabalho em comum com uma ou mais pessoas. Já Cooperação traz o significado mais para auxílio e contribuição.

Segundo Kaye (apud Barros, 1994), colaborar (co-labore) significa trabalhar junto, o que implica no conceito de objetivos compartilhados e uma intenção explícita de somar algo, criar alguma coisa nova ou diferente através da *Colaboração*, contrapondo-se a uma simples troca de informação ou fornecimento de instruções.

No entanto, existe uma discussão quanto ao significado das palavras cooperação e colaboração. Há pesquisadores que acreditam que o termo cooperação é mais abrangente com distinções hierárquicas de ajuda mútua, ao passo que na colaboração existe um objetivo comum entre as pessoas que trabalham em conjunto sem uma hierarquia. Para todos os efeitos, os termos colaboração e cooperação podem ser considerados indistintamente (NITZKE, 1999).

Segundo Malone (apud Borges et al., 1995), cooperação implica em atingir objetivos comuns envolvendo atores diferentes. Desse modo, as aplicações para software cooperativo são direcionadas para efetivar as atividades de grupos de pessoas. De uma forma geral, as ferramentas CSCW estão sendo desenvolvidas para auxiliar as seguintes atividades:

- *brainstorming* para geração de idéias;
- estruturação dessas idéias;
- avaliação das idéias.

2.4. CLASSIFICAÇÕES - TAXONOMIA DAS APLICAÇÕES *GROUPWARE*

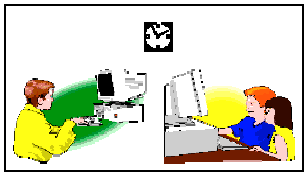
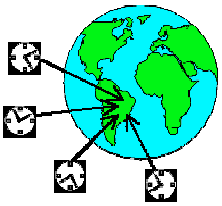
São diversas as classificações propostas para aplicações *Groupware* encontradas nas literaturas, que levam em conta o espaço/tempo, considerando a previsibilidade, entre outros.

2.4.1. CLASSIFICAÇÃO ESPAÇO/TEMPO

Numa primeira abordagem, a taxonomia de sistemas colaborativos é distinguida levando-se em consideração o espaço (onde) e o tempo (quando). Mesmo tempo quando o sistema é síncrono e tempo diferente quando o sistema é assíncrono (SORENSEN, 1995 & BAFOUTSOU et al., 2002).

A tabela a seguir mostra, de forma gráfica, as quatro possibilidades possíveis, considerando espaço e tempo.

TABELA 1 - TAXONOMIA ESPAÇO-TEMPORAL DO GROUPWARE*.*

Taxonomia espaço/tempo	Mesmo tempo	Tempo diferente
Mesmo lugar	Interação face a face 	Interação assíncrona 
Lugar diferente	Interação distribuída síncrona 	Interação distribuída assíncrona 

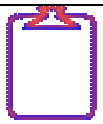
É importante refletir sobre as implicações, do ponto de vista social e técnico, que estão implícitas nessa classificação. No aspecto social há uma enorme diferença entre o encontro físico das pessoas, como numa reunião e uma simulação dessa interação em um ambiente virtual, mesmo que essa ocorra em tempo real (NITZKE, 1999).

2.5. CLASSIFICAÇÃO CONSIDERANDO A PREVISIBILIDADE

Essa classificação, a seguir representada, é citada como sendo uma nova classificação de sistemas de *Groupware,* segundo Nitzke (1999). É sugerido que se leve em conta a possibilidade do lugar e/ou momento no tempo e se são previsíveis ou não. Isso significa que uma atividade pode acontecer dentro de um dado intervalo de tempo (momento previsível ou determinado) ou em um dado local (conhecido ou não).

Citado por Araújo (1995), Grudin (1990) propõe uma nova classificação Espaço/Tempo (tabela 2), incluindo uma categoria intermediária. O autor sugere que seja levada em conta a possibilidade do lugar e/ou do momento no tempo serem previsíveis ou não. Isso significa que uma atividade pode acontecer dentro de um dado intervalo de tempo (momento previsível ou determinado) ou em um dado local (conhecido ou não).

TABELA 2 - TAXONOMIA CONSIDERANDO A PREVISIBILIDADE (NITZKE, 1999).

	Mesmo tempo	Momentos diferentes mas específico	Momentos diferentes e imprevisíveis
Mesmo local	 Auxílio a reuniões	 Deslocamento de tarefas	 Salas de grupos
Diferente mas específico	 Tele/vídeo conferências	 Correio eletrônico	 Edição colaborativa
Diferente e imprevisíveis	 Seminários de interação multicast	 *Bulletin Boards* Eletrônicos	 *Workflow*

No caso do *e-mail*, ao enviar uma correspondência eletrônica, em geral, aguarda-se uma resposta dentro de um tempo razoável, sendo esta uma atividade altamente previsível em relação aos fatores tempo e espaço. Por outro lado, a atividade de escrita colaborativa envolve lugares diferentes, previsíveis e momentos diferentes e totalmente imprevisíveis. Por exemplo,

dois escritores podem realizar a atividade de escrita em locais diferentes, mas cada um em seu local provável, escolhendo, no entanto, momentos totalmente aleatórios para realizar a tarefa (NITZKE, 1999 & BORGES et al., 1995).

2.6. EXEMPLOS DE FERRAMENTAS DE SUPORTE A APLICAÇÕES *GROUPWARE*

Inúmeros são os recursos que podem constituir um *groupware*, como pode-se ver a seguir.

2.6.1. SISTEMAS DE MENSAGENS OU SISTEMAS DE CORREIO ELETRÔNICO

Os sistemas de mensagens suportam a troca assíncrona de mensagens textuais entre grupos de usuários. O correio eletrônico, as listas de interesse, os quadros de aviso (*bulletin boards*) e os *newsgroups* são exemplos desse tipo de aplicação (NITZKE, 1999).

Os sistemas de correio eletrônico são, de fato, gerenciadores de mensagens. A proliferação desse tipo de sistemas, especialmente com a expansão do uso da Internet, juntamente com as listas de interesse, acarretou o que se chama de “sobrecarga de mensagens”. São tantas as mensagens recebidas, que o usuário não consegue ter tempo para processá-las. Por isso, muitos sistemas já incorporam certa “inteligência”, permitindo a classificação automática das mensagens de acordo com seu conteúdo (FARIAS, 2002 & NITZKE, 1999).

2.6.2. SISTEMA DE CO-AUTORIA OU EDITORES COOPERATIVOS

Os editores multi-usuários, ou sistemas de co-autoria, podem ser usados por um grupo para compor e editar um objeto conjuntamente, seja este um gráfico ou um texto. Isso significa que há uma área de trabalho comum onde todos atuam e podem visualizar a atuação dos outros (NITZKE, 1999).

Esse sistema faz com que haja um aumento de eficiência e qualidade no trabalho do grupo, assim como suporta a cooperação entre os autores, durante o desenvolvimento do documento ou gráfico (FARIAS, 2002).

Alguns desses editores não suportam o uso síncrono, sendo mais apropriados para um grupo composto por um editor e vários revisores. Para que um editor multi-usuário possa ser considerado síncrono, este deve oferecer controle de concorrência e mecanismos de atualizações automáticos, isto é, deve ser possível a um usuário, por exemplo, editar uma frase de um parágrafo do texto, enquanto outro está atualizando a frase seguinte, sendo possível que ambos visualizem em tempo real o que o outro está fazendo (NITZKE, 1999).

Os sistemas de co-autoria ou editores colaborativos são utilizados por um grupo para editar um objeto (gráfico ou texto) de forma conjunta. Há editores que suportam o uso síncrono, outros que suportam o uso assíncrono e alguns que trabalham nas duas formas. Os editores assíncronos são mais apropriados para um grupo que tem um editor e vários revisores, que podem apenas visualizar o documento e comentá-lo. Já os editores síncronos, permitem que várias pessoas editem o mesmo documento e visualizem o que os outros estão fazendo em tempo real. Para que um editor seja síncrono, ele deve oferecer controle de concorrência e mecanismo de atualização automático (MACEDO; NETO; PIMENTEL, 2000).

A dificuldade de implementação reside na granularidade do mecanismo de concorrência. Vários sistemas experimentais foram construídos nos últimos anos e experiências realizadas demonstraram a viabilidade do esquema com aumento de produtividade e qualidade (NITZKE, 1999).

Editores colaborativos apoiam o processo de autoria colaborativa de documentos, gráficos ou figuras, pois entendem, nesse caso, que o recurso compartilhado de trabalho é representado pelo documento que está sendo editado em grupo. Nesse documento, as pessoas escrevem suas idéias e os outros componentes do grupo têm acesso.

Alguns editores colaborativos são acoplados a ambientes *CSCW* como um dos mecanismos de comunicação/colaboração, havendo aqueles que são desenvolvidos exclusivamente para a autoria colaborativa (MACEDO; NETO; PIMENTEL, 2000).

2.6.3. SALAS DE REUNIÕES ELETRÔNICAS

As salas de reuniões eletrônicas são sistemas que oferecem ambientes especiais, com grande suporte de *hardware* e *software*, para apoiar reuniões face-a-face. Geralmente envolvem várias estações interligadas em rede, o uso de telões computadorizados e de equipamentos de áudio e de vídeo.

2.6.4. CONFERÊNCIAS ELETRÔNICAS

O computador serve como um meio de comunicação em uma grande variedade de formas. A conferência em uma rede corporativa permite que diversos usuários reunidos em uma sala de reunião eletrônica, ou dispersos fisicamente, interajam sincronizados através de suas estações de trabalho.

2.6.5. SISTEMAS DE WORKFLOW

Os sistemas de *Workflow* são definidos como uma automação de processos de um negócio. Esses sistemas podem compreender uma parte desse negócio ou todo o sistema, onde, cada documento, informações e tarefas são passados para cada participante do grupo, de acordo com regras e procedimentos sistematizados (FARIAS, 2002).

Tendo como objetivo fornecer ajuda aos usuários na realização de suas tarefas, os sistemas de gerenciamento de *Workflow* coordenam as ações desses usuários, direcionando ações corretas, num momento exato em que devem ocorrer, para o cumprimento das atividades, baseado no modelo do trabalho cooperativo que está inserido no contexto dos sistemas de *workflow* (HOFTE, 1998).

Pode-se também definir o sistema de *Workflow* como um modelo empresarial computadorizado que especifica todos os parâmetros envolvidos em sua realização. É também definido como uma descrição computadorizada, inter-operável de atividades e sua ordem de execução (SILVA FILHO; WAINER; MADEIRA, 2002).

O Sistema de Gerenciamento de *Workflow*, consiste em um conjunto de aplicações de controle e interfaces para outras ferramentas e aplicações que permitem o seu projeto, definição, execução e monitoramento. (FARIAS, 2002).

Nesse tipo de sistema, os documentos são compartilhados por várias pessoas geograficamente distribuídas e passam seqüencialmente, de uma pessoa para outra, sendo estas responsáveis por adicionar ou refinar informações nestes documentos.

2.6.6. APLICAÇÕES CHAT

Similares ao *e-mail*, é uma das aplicações mais populares em *Groupware*. Essas aplicações permitem que os usuários troquem mensagens entre si, tanto de forma síncrona, como assíncrona.

2.6.7. QUADRO BRANCO COMPARATIVO (SHARED WHITEBOARDS SYSTEM)

Essas aplicações são utilizadas principalmente em reuniões que envolvem discussão em torno do desenho de um projeto, onde as pessoas envolvidas manipulam as formas apresentadas, inserem novos desenhos, etc.

Os sistemas de quadro branco compartilhados foram desenhados especificamente para fornecer suporte a reuniões, especialmente quando os integrantes estão geograficamente dispersos e necessitam exercer uma atividade cooperativa.

2.6.8. LOTUS NOTES

Lotus Notes R5 é um ambiente integrado de mensagens e colaboração como a Web, que oferece aos usuários um acesso rápido e gerenciamento de muitos tipos de informações, incluindo correio eletrônico, compromissos, contatos, lista de tarefas e páginas da Web, Grupos de Notícias e aplicações intranet baseados na Web. O Notes agrupa todas essas informações em um único ambiente. Fácil de usar, personalizável e seguro, oferece poderosas inovações no gerenciamento de mensagens, colaboração e informações.

O *Notes* possibilita a seus usuários gerarem documentos e organizá-los em bases de dados, de forma que todos possam encontrar rapidamente as informações que necessitam. Por permitir a organização de uma grande quantidade de documentos e informações gerados por diversos indivíduos, o *Notes* auxilia no monitoramento e gerenciamento das atividades e produção do grupo de trabalho, mas é especialmente próprio para o desenvolvimento de sistemas de *Workflow* (BORGES; CAVALCANTI; CAMPOS,1995).

2.7. SUPORTE COMPUTACIONAL PARA TRABALHOS COOPERATIVOS

O bom entendimento do trabalho cooperativo é de vital importância antes de se construir um suporte para o *CSCW*. O contexto do trabalho cooperativo é caracterizado pelas tarefas que as envolvem, pelas pessoas que estão inseridas na execução dessas tarefas e como essas tarefas afetam a outros (FARIAS, 2002).

O suporte computacional deve refletir a necessidade das pessoas, ou seja, de como elas colaboram. Esse suporte deve possibilitar uma infra-estrutura tecnológica para facilitar a execução de tarefas colaborativas, enquanto mantém os benefícios de interação social das pessoas envolvidas. Além disso, para algumas tarefas, pode ser desnecessário e até indesejável quando inibe as interações sociais entre os indivíduos (FARIAS, 2002).

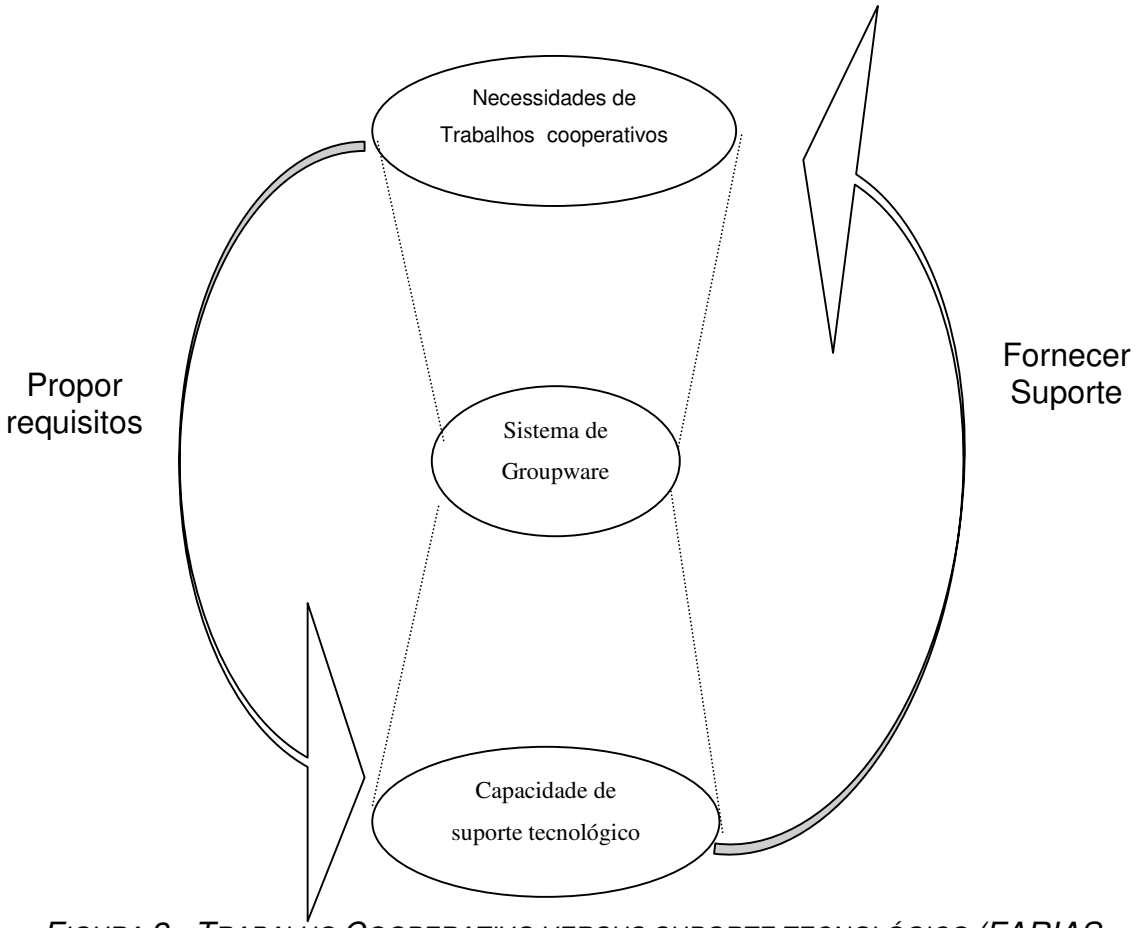


FIGURA 2 - TRABALHO COOPERATIVO VERSUS SUPORTE TECNOLÓGICO (FARIAS, 2002)

A figura 2 ilustra o relacionamento entre o trabalho cooperativo e o suporte computacional. De um lado, o trabalho cooperativo tem necessidade de propor uma série de requisitos para o suporte tecnológico, que, por sua vez deve fornecer o suporte (soluções). Por outro lado, a Capacidade de Suporte Tecnológico provê os tipos de funcionalidades para suportar tais necessidades. Como resultado, tem-se um ambiente no qual os sistemas de *Groupware* são desenvolvidos com um compromisso entre o que deveria ser feito e o que pode ser feito para que o trabalho cooperativo se realize adequadamente.

2.8. PLATAFORMAS DE DESENVOLVIMENTO DOS SISTEMAS DE *GROUPWARE*

Uma das principais preocupações no desenvolvimento dos sistemas de *Groupware* está relacionada com a reutilização de funcionalidades e do suporte *run-time*, ou seja, não desenvolver novamente o que já está

consolidado como função, já que nos dias iniciais dos sistemas *Groupware* já foram desenvolvidos manualmente, ou seja, codificados no seu todo (FARIAS, 2002).

A seguir, descrever-se-ão três plataformas nas quais os sistemas de *Groupware* foram desenvolvidos nos meados dos anos 90, que também são chamados de *Groupware Toolkit* (FARIAS, 2002).

GroupKit é uma plataforma usada no desenvolvimento dos sistemas de *Groupware* em tempo-real (*Real-time*). Uma das características do *GroupKit* é o uso de uma arquitetura semi-replicada, que contém três importantes componentes, conforme descrito a seguir:

- **registrar**: é um ponto centralizador dos pontos de conexão de uma conferência;

- **gerenciamento de sessão**: é um processo replicado que fornece suporte para criar, apagar, associar e abandonar a conferência. Esse processo provê não somente uma interface de uso, mas também uma política de gerenciamento de conferência;

- **aplicação para conferência**: é também um processo replicado que fornece as funcionalidades para as aplicações de *Groupware* simples, tais como: W*hiteboard* compartilhado ou um sistema de *Chat;*

- **COAST** é uma plataforma O*bject-Based* usado no desenvolvimento de aplicação de tempo-real (*Real-Time*). Utiliza uma abordagem de arquitetura replicada, em que cada local da aplicação tem uma cópia do documento original que é compartilhado. A aplicação do usuário interage com o documento via V*iew Object*, C*ontroller Object* e *Session Object*.

- ***COLA***, é uma outra plataforma *Object-based,* usada no desenvolvimento de sistema de *Groupware* síncrono. *COLA* é baseado numa arquitetura distribuída em que um modelo de

atividade leve é usado para prover mecanismo para descrever o contexto da cooperação ou situação baseada em objetos compartilhados.

- ***JAZZ,*** é um projeto de pesquisa da IBM que está inserida no contexto colaborativo, cuja a finalidade é proporcionar um ambiente de desenvolvimento de aplicações colaborativas. Está baseado na filosofia de desenvolvimento de software "*open office*" (HUPFER, 2004).

2.9. ADAPTABILIDADE EM *GROUPWARE*

Um sistema de *Groupware* é normalmente desenvolvido para suportar um controle específico de trabalho em grupo. No caso de mudanças no contexto desse trabalho, o sistema deve possibilitar as adaptações necessárias para refletir as novas mudanças (FARIAS, 2002).

Conforme ilustra a figura 3, quando ocorre a mudança do contexto do trabalho cooperativo, o sistema de *Groupware* deve ser adaptado, gerando um novo sistema para o novo contexto desse novo trabalho a fim de suprir os novos requisitos.

Segundo Farias (2002), atualmente, a adaptabilidade é largamente reconhecida como um requisito central no desenho de sistemas *Groupware.* É definida como a atividade de modificar um sistema computacional dentro do contexto de uso. Pode-se ver a seguir alguns níveis da adaptabilidade:

- customização: opção de configuração – personalização;
- integração: Trata-se da composição de parâmetros pré-definidos dos componentes para unificação do sistema;
- extensão: adicionar novas funcionalidades do sistema como um simples acoplamento de novos componentes.

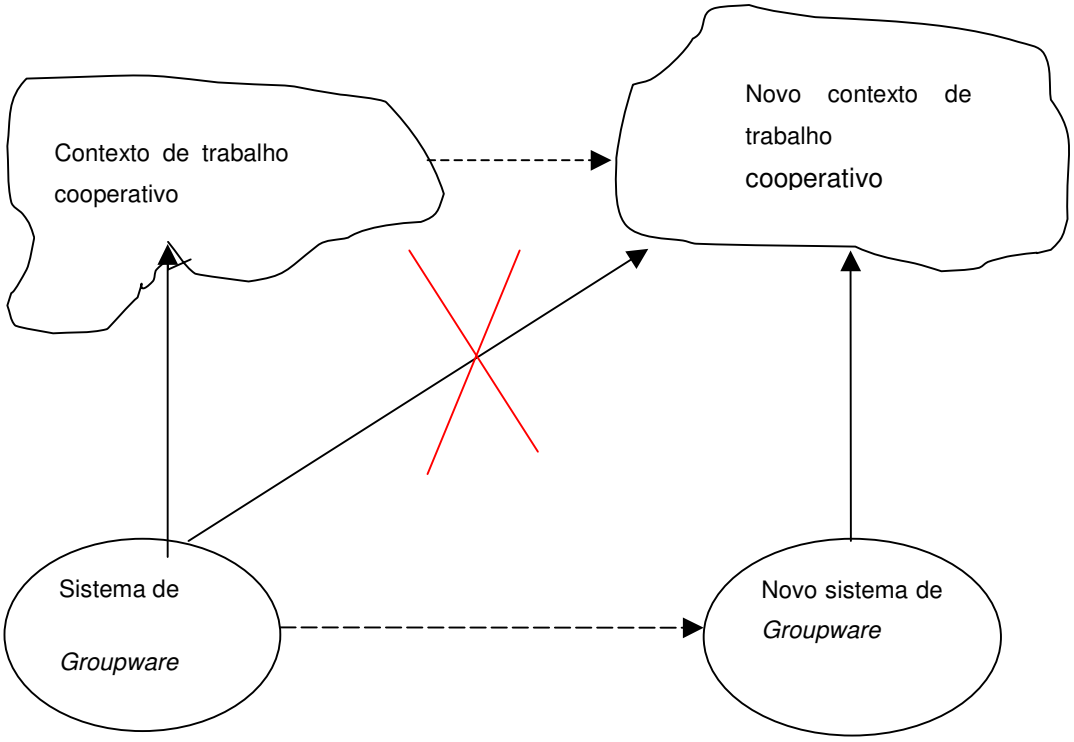


FIGURA 3 - ADAPTAÇÃO DE SUPORTE DO TRABALHO COOPERATIVO (FARIAS, 2002)

2.10. *GROUPWARE* NO CONTEXTO DE SISTEMAS DISTRIBUÍDOS

Segundo Tanenbaum (2000), Sistemas Distribuídos consistem em uma coleção de computadores autônomos ligados por uma rede de comunicação e equipados com um software de sistema distribuído. O sistema distribuído exerce um papel no qual permite que os computadores coordenem suas atividades e compartilhem seus recursos, tais como hardware, software, dados etc.

O *Groupware* é considerado um Sistema Distribuído em Tempo Real, pois permite que duas ou mais pessoas separadas geograficamente, trabalhem de forma conjunta e ao mesmo tempo em ambientes computadorizados. Esse por sua vez, suporta a grupos de habilidades para manipular artefatos (documentos) em um espaço de trabalho compartilhado e que estão distribuídos (ou replicados) para cada participante do grupo de trabalho (GREENBERG; MARWOOD, 1994).

2.11. CARACTERÍSTICA DO SISTEMA DISTRIBUÍDO

Para Tanenbaum (2000), os sistemas distribuídos têm crescido nas últimas décadas em relação aos sistemas centralizados, principalmente pelo barateamento de recursos como hardware, pelo advento dos microprocessadores com alta tecnologia, redes com alta capacidade de tráfego etc.

Um sistema distribuído pode ter um poder de processamento maior que o de um sistema centralizado, aliado ao fator confiabilidade, pois se uma máquina sair do ar, o sistema como um todo pode sobreviver, bem como proporcionar um crescimento de forma incremental à medida da necessidade de seus usuários.

Segundo Bhushan (1998), já no contexto das aplicações colaborativas, as características dos sistemas distribuídos se fazem necessários, pois seus recursos estão espalhados, em qualquer parte da rede e que há necessidade dessas aplicações compartilharem seus dados.

Outra característica inerente ao trabalho em grupo, é que muitos dos usuários podem estar trabalhando em locais diferentes e, consequentemente, trabalhando com recursos, computadores, em locais diferentes e interligados.

Apesar das características positivas do sistema distribuído em relação ao centralizado, Tanenbaum (2000), considera que há alguns fatores que devem ser considerados como agravantes e que despertam atenção quando são aplicados.

Outros aspectos importantes dos sistemas distribuídos que devem ser considerados para um projeto são: a transparência, flexibilidade, confiabilidade, o desempenho e a escalabilidade.

(a) Transparência

A transparência tem pôr objetivo passar uma imagem de que o sistema é único, ou seja, apesar de vários hardwares e softwares envolvidos, tem-se um conceito único de ambiente, uma coisa só.

A transparência pode ser aplicada:

- transparência à localização, quando os usuários não devem saber onde os recursos se encontram;
- transparência na migração, quando os recursos podem trocar de lugar à vontade sem ter que mudar de nome;
- transparência na replicação, quando os usuários não devem saber quantas cópias existem;
- transparência na concorrência, quando vários usuários podem compartilhar automaticamente os recursos;
- transparência no paralelismo, quando podem ocorrer atividades paralelas sem que os usuários venham a conhecer.

(b) Flexibilidade

A segunda característica importante do projeto de um sistema distribuído é a flexibilidade. É muito importante que o sistema seja flexível, pois qualquer necessidade não prevista ou mesmo correção que se possa fazer deve ser executada sem maiores prejuízos para o sistema.

Há duas escolas de pensamento a respeito da estrutura dos sistemas distribuídos: sendo uma baseada em funções centralizadas (*Kernel* monolítico) com alta performance, porém menos flexível; a outra linha sustenta que cada máquina deve possuir seu próprio *Kernel* tradicional e que forneça a maioria dos serviços a ele solicitado (*MicroKernel*). Esse último é mais flexível, e apesar do domínio do *Kernel* monolítico, vem ganhando espaço.

Basicamente ele fornece quatro serviços:

- um mecanismo de comunicação de processos;
- um mínimo de funções para gerência de memória;
- um mínimo de funções de gerência de processos e de escalonamento;
- funções de entrada/saída de baixo nível.

(c) Confiabilidade

Os sistemas distribuídos devem ser mais confiáveis do que os sistemas centralizados, pois quando há interrupção no funcionamento de uma das máquinas, uma outra máquina passa a assumir os serviços realizados por aquela que parou. Existem alguns servidores, em diferentes máquinas, que precisam estar no ar para que o sistema funcione como um todo.

Os aspectos relacionados à confiabilidade que devem ser levados em consideração são:

- **disponibilidade**: refere-se à fração de tempo em que o sistema está funcionando. Com isso, pode-se melhorar a disponibilidade através de um projeto que não exige o funcionamento simultâneo de um número substancial de componentes críticos, ou seja, ter outras ferramentas de substituição, tais como a redundância. Isso significa que componentes chaves de hardware e software devem ser replicados, de modo que se um deles vier a falhar, os outros estarão disponíveis para assumir o papel de execução das tarefas.
- **tolerância a falhas**: quando um dos servidores falhar será possível mascarar as falhas, isto é, não permitem que o usuário perceba o que está acontecendo. Um exemplo clássico é o de serviço de arquivos, que pode ser construído com um grupo de servidores cooperantes, de tal forma que o usuário não perceba a falta de um ou mais servidores, apenas "estranhe" o desempenho.

(d) Desempenho

A construção de um sistema distribuído que seja transparente, flexível e confiável não terá muito sentido se o aspecto de desempenho e tempo de resposta à solicitação, for lento, ou seja, com alto tempo de resposta.

Uma aplicação que funciona em sistema distribuído deve ser superior em desempenho do que a mesma aplicação que é executada em sistema com um único processador. E, para medir e apurar esse diferencial podem ser usadas várias métricas de apuração, tais como:

- tempo de resposta;
- *throughout* (números de tarefas por hora);
- utilização do sistema;
- quantidade de recursos da rede.

O problema de desempenho é fortemente influenciado pela comunicação, que é uma essência do sistema distribuído.

(a) Escalabilidade

Os sistemas distribuídos precisam adaptar-se à possibilidade de existirem ambientes com milhares de processadores. Para isso deve-se evitar :

- componentes centralizados: um único servidor para todos os usuários;
- tabelas centralizadas: uma única tabela de endereço de localização de arquivo;
- algoritmos centralizados;

Para ambientes distribuídos deve-se usar algoritmos descentralizados que possuam as seguintes características:

- nenhuma máquina deve centralizar informações sobre o estado do sistema;
- as máquinas devem tomar decisões baseadas apenas nas informações disponíveis localmente;
- se uma das máquinas falhar não irá impedir o funcionamento do algoritmo;
- não há suposições implícitas que exista um relógio global.

2.12. SISTEMA DE GERENCIAMENTO COOPERATIVO

O conceito e técnica de trabalho cooperativo em *CSCW* têm sido utilizados em várias áreas computacionais, tais como: multimídia, sistema de suporte a decisão, e várias outras áreas. Mas, raramente, são aplicados em um contexto de sistema de gerenciamento. Informações compartilhadas e comunicação são considerações chaves para o trabalho cooperativo (BHUSHAN, 1998).

Tem como objetivo o trabalho cooperativo para proporcionar um número de comunicação com entidades, tais como: processos, aplicações, e pessoas, de forma conjunta, para resolver de forma eficiente um problema em comum, tomar decisões, dentre outras atividades. Para o conceito de Sistemas Distribuídos, no qual os recursos estão espalhados por toda a rede (Tanenbaum, 2000), há necessidade de se ter o sistema de gerenciamento cooperativo (BHUSHAN, 1998).

Os requisitos significativos que são levados em consideração na definição de um sistema de gerenciamento cooperativo são determinados conforme a ISO que incluem os aspectos de sistemas distribuídos, heterogeneidade de problemas e conceito de gerenciamento de sistema (BHUSHAN, 1998).

As áreas chaves de gerenciamento funcional são: Falhas (*Fault*), Configuração (*Configuration*), Estatística (*Accounting*), Desempenho (*Performance*) e Segurança (*Security*) representada pela sigla (FCAPS). Em combinação, essas áreas têm sido aceitas como um meio para descrever os requisitos de qualquer sistema de gerenciamento (BHUSHAN, 1998).

2.13. PROBLEMAS PERTINENTES AOS SISTEMAS DISTRIBUÍDOS

Uma das preocupações quando se menciona Sistemas Distribuídos são os aspectos da comunicação e da sincronização. Esses dois aspectos estão intimamente ligados, pois os processos (aplicações) só conseguem cooperar com a sua presença (TANENBAUM, 2000).

Um dos problemas relativos aos sistemas distribuídos, que faz com que a sincronização falhe, está relacionado com o fator tempo, pois desempenha um papel importantíssimo em qualquer método de sincronização (TANENBAUM, 2000).

Vários algoritmos têm sido propostos, tais como o de *Lamport* (1978,1990), algoritmo para sincronização de *Clock*, algoritmo de *Cristiam* que utiliza um máquina como servidor de tempo para responder as requisições, dentre outros. Tais algoritmos trazem alguns métodos para solucionar esse problema (TANENBAUM, 2000).

Além do fator tempo, outro aspecto a ser considerado é o controle de concorrência (Greenberg & Marwood, 1994), ou seja,

> "...quando várias transações estiverem sendo executadas simultaneamente por diferentes processos, rodando em processadores diferente, há necessidade de algum mecanismo para que nenhum deles interfiram no processamento dos demais" (TANENBAUM, 2000).

O *Deadlock* é um outro problema que precisa ser tratado no contexto de Sistema Distribuído. Pode ocorrer quando pelo menos duas aplicações tentam alocar o mesmo recurso, provocando o travamento da aplicação. Alguns algoritmos também são propostos a fim de prevenir ou

detectar o *Deadlock*, tais como: o algoritmo do avestruz, Detecção de *Deadlock* Distribuído, Prevenção de *Deadlock* Distribuídos (TANENBAUM, 2000).

Pode ser fácil conceituar um espaço compartilhado, com muitos usuários trabalhando, como nos ambientes constituídos fisicamente, e, espera-se ver imediatamente aquilo que os outros estão fazendo. Porém, caso não sejam tomados os devidos cuidados, um sistema de *Groupware* poderá sofrer as conseqüências do controle de concorrência, eventos chegando fora de ordem, deixando, por exemplo, uma imagem ou documento inconsistente, a nível de base de dados e um efeito desagradável e indesejável na tela (GREENBERG; MARWOOD, 1994).

Porém, diferente dos sistemas distribuídos tradicionais, que não podem tolerar a inconsistência das bases de dados, o sistema de *Groupware* pode ser mais ou menos tolerante a esse tipo de ocorrência, pois o próprio usuário poderá estar mediando e reparando eventuais inconsistências (GREENBERG; MARWOOD, 1994).

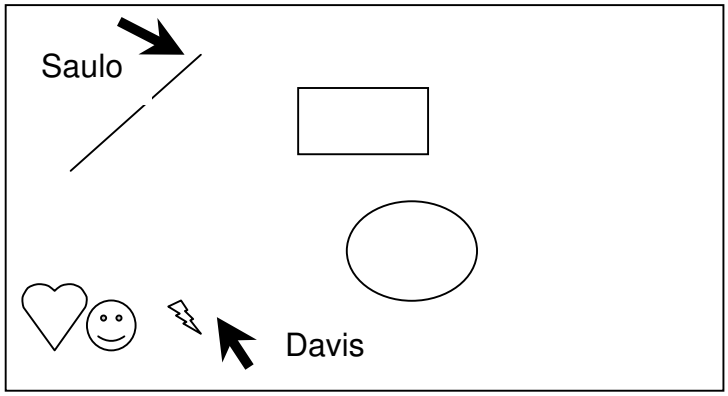


FIGURA 4 - DOIS USUÁRIOS EM UMA FERRAMENTA DE DESENHO EM GRUPO (GREENBERG; MARWOOD, 1994).

A figura 4, exemplifica uma ferramenta de aplicativo de desenho em grupo, que permite que cada membro do grupo crie, manipule e

edite objetos, tais como: figuras, linhas, formas geométricas etc, sem o controle de concorrência.

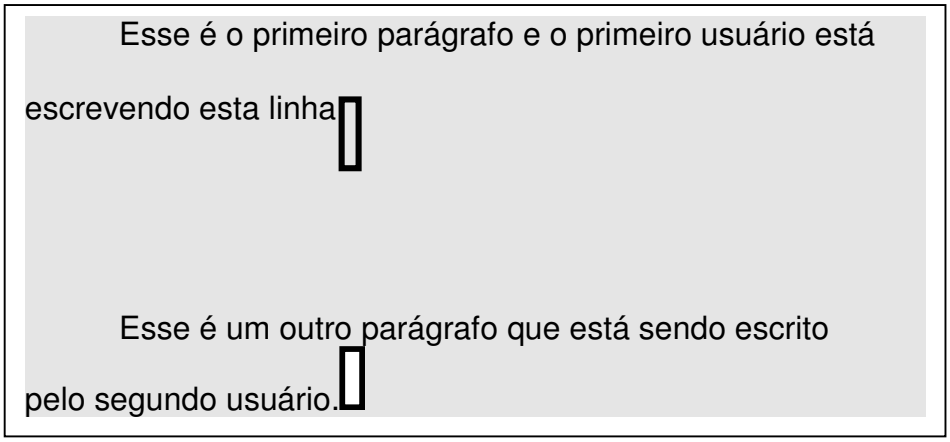


FIGURA 5 - DOIS USUÁRIOS EM UM EDITOR DE TEXTO COLABORATIVO (GREENBERG; MARWOOD, 1994).

A figura 5, considera um editor de texto *Groupware* que permite muitos usuários entrarem e manipularem o texto ao mesmo tempo

Nas aplicações anteriores, a ocorrência de inconsistências não tem muita importância, pois pode ser corrigida pelos próprios integrantes do grupo de trabalho.

A Serialização[4] ou o *Locking* podem ser uma das formas de garantir que outros usuários acessem o mesmo objeto em um ambiente colaborativo ao mesmo tempo. Porém, traz consigo a perda de performance da aplicação, razão pela qual muitas das aplicações colaborativas não são implementadas, além do fator custo. Caso, porém, seja adotado um método de controle de concorrência, a escolha tem que ser bem feita, pois uma escolha errada pode inviabilizar o sistema (GREENBERG; MARWOOD, 1994).

[4]Serialização – As transações concorrentes não podem interferir uma com as outras. Assegura que se duas ou mais estiverem rodando ao mesmo tempo, para cada uma delas e para os outros processos, o resultado final aparecerá como se as transações rodassem seqüencialmente em alguma ordem pré-definida dependente do sistema (TANENBAUM, 2000).

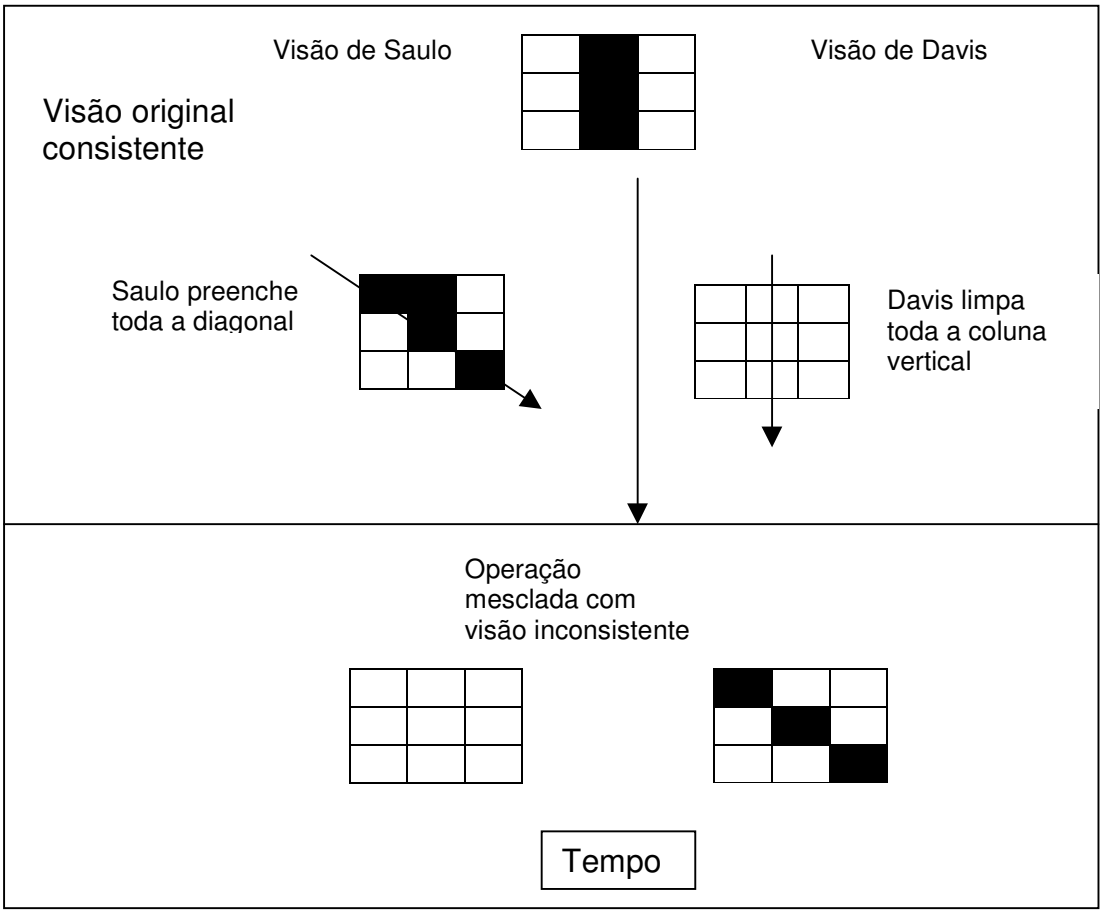


FIGURA 6 - DOIS USUÁRIOS EDITANDO UMA MESMA FIGURAS GEOMÉTRICAS EM UMA APLICAÇÃO COLABORATIVA (GREENBERG; MARWOOD, 1994).

Na figura 6, a falta de controle de concorrência produzirá um resultado indesejável, já que ambos os usuários estão manipulando o mesmo objeto e simultaneamente realizando mudança que se sobrepõem às ações de outros usuários. Nesse caso, o controle de concorrência seria bem-vindo.

3. FERRAMENTAS CASE PARA *CSCW*

A ferramenta *CASE (Engenharia de Software Auxiliada por Computador),* segundo Damm (2000), é usada para auxiliar no projeto e desenvolvimento de um *software*, fornecendo suporte para a modelagem dos requisitos do sistema, a assistência automatizada para o desenvolvimento, com geração do código fonte e documentação. Conforme Lending e Chervany (1998), abrange também muitos produtos com diversas funcionalidades. Segundo Martin (1991), supre a fase de análise com facilidades gráficas para o planejamento e projeto de sistema. São essenciais para a obtenção de apoio automatizados para o projeto de sistema, verificação computadorizada do projeto e geração de código a partir do projeto.

A introdução de uma ferramenta *CASE* no desenvolvimento de *software* tem por objetivo:

(a) reduzir o tempo e custo do desenvolvimento (LENDING; CHERVANY, 1998);

(b) aumentar a qualidade do desenvolvimento (LENDING; CHERVANY, 1998);

(c) direcionar os desenvolvedores a trabalhar dentro de padrões e metodologias (DAMM, 2000).

São as seguintes algumas características de uma ferramenta *CASE* (DAMM, 2000 & LENDING et al., 1998):

- criar e editar diagramas, tais como DFD, MER etc;
- armazenar atributos e informações dos diagramas e testar os modelos construídos;
- compartilhar e combinar diagramas de diferentes usuários;

- gerar códigos a partir dos diagramas ou produzir diagramas a partir dos códigos; e

- documentação do sistema.

Há pelo menos três tipos de classificação das ferramentas CASE: (1) *UPPER CASE* (*front end CASE*), que fornece suporte para os estágios iniciais do ciclo de vida do sistema, como a análise de requisitos e projeto; (2) *LOWER CASE* (*back end CASE*), que fornece suporte para os estágios finais do ciclo de vida do sistema (geração de código e teste); (3) *Integrated CASE* (front and back end), que fornece suporte a todas as fases do desenvolvimento de sistema (LENDING; CHERVANY, 1998).

Quanto à filosofia de aplicação de controle de metodologia, pode-se classificar em 3 níveis: (1) Ferramenta *CASE* restritiva, que executa uma abordagem "*TOP-DOWN*"; (2) Ferramenta *CASE* dirigida, que sugere mas não obriga a execução "*TOP-DOWN*"; (3) Ferramenta *CASE* flexível, na qual o desenvolvedor tem total controle no desenvolvimento do processo (VERSEY, 1995).

3.1. PESQUISA DE USO DAS FERRAMENTAS *CASE*

Um estudo de implementação de ferramentas *CASE* realizado em 100 (cem) organizações da Dinamarca e Finlândia, demonstrou que, geralmente, seus usuários estavam satisfeitos com as facilidades fornecidas, porém, desapontados pelo fato destas não suportarem múltiplos usuários, ou seja, ambientes colaborativos (SORENSEN, 1995).

As ferramentas *CASE* são fracas para suportarem criatividade, geração de idéias e resolução de problemas (DAMM et al., 2000).

Outros estudos revelaram que o uso de ferramentas *CASE* é limitado. Das 53 companhias desenvolvedoras de software pesquisadas, 39 (73,5%) nunca usaram ferramentas *CASE*. Das 14 companhias que tentaram sua utilização, 5 delas abandonaram as ferramentas. Os entrevistados de 14

companhias acreditam que o uso das ferramentas *CASE* aumenta a qualidade da documentação, melhora a análise e os sistemas tornam-se mais fáceis de serem testados e receberem manutenção. Entretanto, também apontaram que as ferramentas *CASE* são de difícil utilização e demandam tempo.

Em outras pesquisas, somente 24% das companhias estavam usando as ferramentas *CASE*. Continuando as pesquisas, dos gerentes entrevistados, 13 (treze), disseram que já tinham utilizado ferramentas *CASE* há pelo menos dois anos atrás e 4 (quatro) disseram que continuam utilizando a ferramenta *CASE*. A razão para o abandono, inclui o custo (em torno $22.000,00 por pessoa, incluindo software, hardware e treinamento), falha nos retornos estimados e expectativas irrealistas (LENDING; CHERVANY, 1998).

Das pesquisas realizadas, duas razões apontam para o baixo uso das ferramentas *CASE*: (1) o trabalho que os desenvolvedores de sistema fazem não é o mesmo quando uma ferramenta *CASE* é utilizada, ou seja, pela própria característica dessas ferramentas, de adotarem uma metodologia, são totalmente formais, enquanto que a tendência dos desenvolvedores está direcionada para a informalidade, sendo que o tempo "gasto" com a ferramenta *CASE* é maior nas fases iniciais, mas compensado nas fases seguintes do tempo dos desenvolvedores. Além desses fatores, a preferência dos desenvolvedores é trabalhar com alta autonomia no desenvolvimento; (2) os desenvolvedores não aparentam motivos intrínsecos[5] e motivos extrínsecos (potencialidade de uso).

Em pesquisas anteriormente sobre o uso de ferramentas *CASE* apontaram que: (1) poucas organizações usam ferramentas *CASE*; (2) (é comum) as organizações abandonarem o uso das ferramentas; (3) as organizações que usam ferramentas *CASE* possuem muitos desenvolvedores de sistemas que, na prática, não usam a ferramenta (LENDING, CHERVANY, 1998).

[5] Intrínseco, segundo o dicionário Aurélio: "Que está dentro de uma coisa ou pessoa e lhe é próprio; interior, íntimo:"

3.2. NECESSIDADE DAS FERRAMENTAS SEREM COLABORATIVAS

Um dos fatores dessa necessidade, segundo Sorensen (1995), é que muitos projetos de engenharia de software são complexos para que uma única pessoa gerencie e, na prática, ocorre a colaboração entre os desenvolvedores.

Conforme Versey (1995), as ferramentas atuais disponíveis no mercado são projetadas, em sua maioria, para usuários simples (ferramentas mono-usuárias).

Sistemas complexos requerem a colaboração em diferentes aspectos do trabalho em conjunto num período grande de tempo para a criação e conclusão do produto final (software). Os membros dos grupos de trabalho compartilham informações, realizam tarefas independentes e criam produtos de identidade comum, ou seja, um produto que teve a participação conjunta de todos os membros da equipe.

No ambiente de coletividade (Colaboração), a capacidade de coordenação do trabalho em grupo, na solução de problemas simples, ou mesmo múltiplos grupos trabalhando em múltiplos problemas, é um fator crítico de sucesso, no que se refere ao projeto final. As ferramentas, principalmente as *CASE* que se propõem a suportar um grupo de trabalho (Colaboração) no desenvolvimento de sistemas, devem dar suporte às características como a coordenação de atividades integradas e gerenciamento do projeto (VERSEY, 1995).

Outro fator importante, quando ocorrem mudanças no projeto (diagramas, especificações, documentação e código fonte), produzidas pelo grupo, devem ser controladas e coordenadas de forma eficiente e eficaz poupando recursos do projeto (SORENSEN, 1995).

Como ocorre na maioria dos trabalhos de desenvolvimento de software, segundo Versey (1995), 70% a 80% do tempo é dedicado ao trabalho em equipe e muitos dos recursos são consumidos para manter um controle

mínimo de comunicação e gerenciamento (SORENSEN, 1995 & VERSEY, 1995).

Uma equipe heterogênea ou interdisciplinar não pode trabalhar com boa produtividade e de forma eficaz num projeto complexo, a menos que esse trabalho seja amparado por uma metodologia específica para esse fim. Isso tende a ser um fator agravante de maior intensidade quando um projeto é de larga escala.

Os maiores fatores de falhas apontados por pesquisadores dão-se pela pobre comunicação entre os usuários, desenvolvedores e gerentes. Mas, em contraste, os sistemas CSCW fazem aumentar a comunicação pelo próprio contexto do ambiente colaborativo (BAILEY; SWIGGER; VANECEK, 1995).

Em pesquisas realizadas sobre as ferramentas *CASE*, foram apontados alguns requisitos básicos para que tivessem o aspecto da colaboração (CSCW), (SORENSEN, 1995 & FARIAS, 2002 & BAILEY et al., 1995):

- modelos compartilhados e distribuídos;
- *merging* dos modelos;
- administração, estatística e relatórios de análises e impactos;
- controle de versões;
- múltiplos níveis de segurança;
- manutenção dos diálogos entre os membros da equipe;
- permitir que a equipe trabalhe simultaneamente em uma tarefa simples;
- gerenciamento de email;
- uso concorrente de dicionários, diagramas, etc;

- suporte de interação de grupo (*Brainstorming*);
- dispositivo para notificar os grupos quando ocorrer mudança nas atividades;
- repositório e enciclopédia de cada fase dos produtos desenvolvidos.

Os requisitos de coordenação e distribuição são parte inerente a uma ferramenta que se propõe a formar trabalho em grupo (SORENSEN, 1995 & BAILEY et al., 1995).

Um requisito importante para uma ferramenta CASE é o desenvolvimento de uma arquitetura de suporte colaborativa no intuito de atingir os objetivos da padronização e reutilização de componentes. Essa arquitetura deve satisfazer os seguintes critérios: (1) A arquitetura deve ser modular, ou seja, pode ser desenvolvida e implementada em fase; (2) deve ser genérica para ser compatível com qualquer ambiente (VERSEY, 1995).

Uma arquitetura de suporte a colaboração requer pelo menos três níveis, como ilustra a Figura 3: (1) *Taskware*, que é uma camada básica que possibilita a execução de tarefas que não precisam ser compartilhadas; (2) *Teamware*, é uma camada intermediária, onde é necessário que algumas tarefas sejam compartilhadas e seus resultados também sejam compartilhados com outros membros do grupo; (3) *Groupware*, é uma camada superior que suporta tarefas independentes (aplicativos como *e-mail,* boletim, calendários), de necessidade de um grupo trabalho, como por exemplo, comunicação direta com outros membros de um grupo.

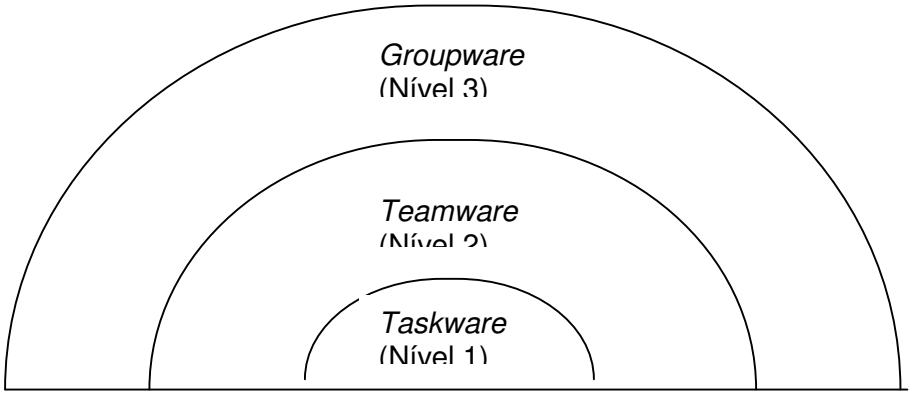


FIGURA 7 - MODELO ARQUITETURAL DE SUPORTE COLABORATIVO (VERSEY, 1995).

O objetivo do *Taskware* é suportar a condução de um trabalho em particular, enquanto os objetivos do *Teamware* e *Groupware* são de suportar trabalho em grupo num interesse de completar um trabalho e desenvolver o trabalho final que se projetou (VERSEY, 1995).

Os modelos representados nas figuras 4, 5, 6, representam os requisitos envolvidos na elaboração de uma ferramenta CASE colaborativa:

(1) *(CSCW)* que diz respeito à realização da execução de atividades em grupo;

(2) *(SEE); (3) DataBase;* e (4) CASE (VERSEY, 1995). Os principais requisitos são para coordenação e cooperação (VERSEY, 1995).

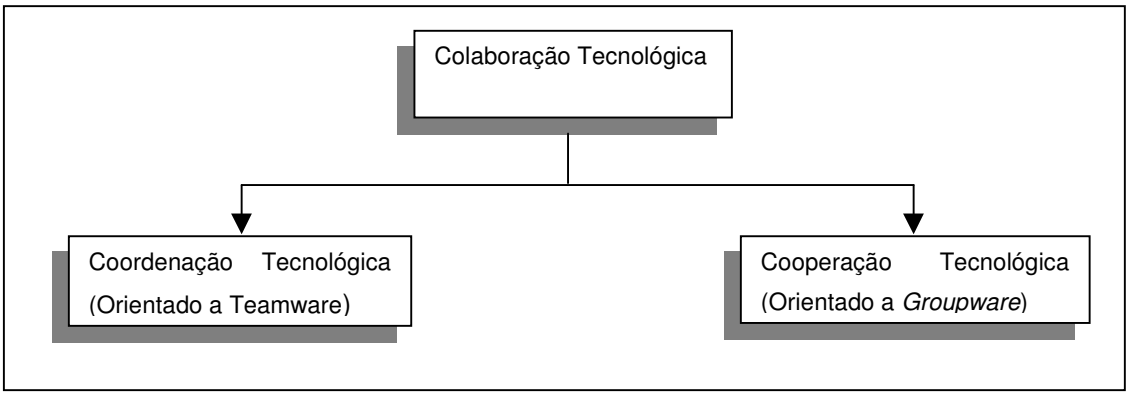


FIGURA 8 - MODELO DE COLABORAÇÃO PARA SUPORTE DE AMBIENTE (VERSEY, 1995).

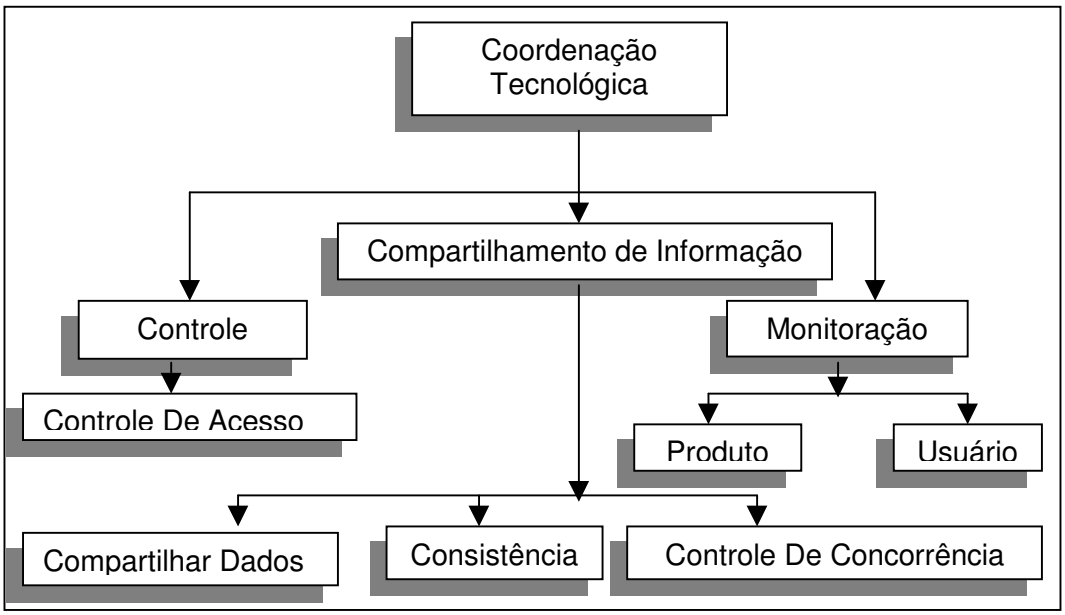


FIGURA 9 - MODELO DE COORDENAÇÃO PARA SUPORTE DE AMBIENTE (VERSEY, 1995).

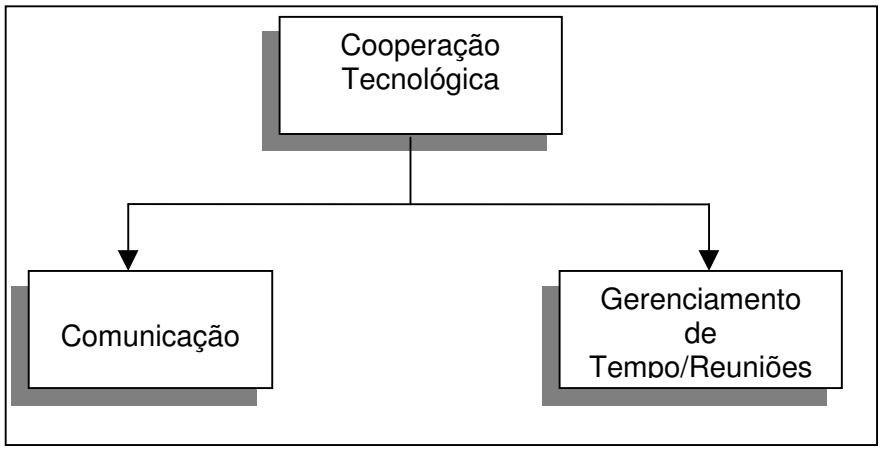


FIGURA 10 - MODELO DE COOPERAÇÃO PARA SUPORTE DE AMBIENTE (VERSEY, 1995).

Enquanto o papel do *Teamware* é, principalmente, o da coordenação de atividades dos membros do grupo e a coordenação que tem um papel organizacional, o papel do *Groupware* é suportar o trabalho em grupo no aspecto da cooperação, que é tipicamente suprir a necessidade de comunicação (VERSEY, 1995).

3.3. REQUISITOS ADICIONAIS DE UMA FERRAMENTA COLABORATIVA

Muitos esforços têm sido feitos para o desenvolvimento de sistemas de *Groupware* síncronos que, além de fornecer comunicação entre os membros de grupo, também fornecem informações a respeito de cada membro presente e suas ações (FARIAS, 2002).

Um dos problemas que permeiam as aplicações de *Groupware*, segundo Farias (2002), é a dificuldade de gerenciar as consistências, ou seja, evitar que as informações produzidas por integrantes sejam inconsistentes entre si. Para isso, alguma abordagem deve ser adotada para se manter a consistência ou gerenciar as inconsistências quando ocorrer.

Apesar das diversas facilidades, as ferramentas *CASE* não são amplamente utilizadas. Segundo Damm (2000), as ferramentas *CASE* normalmente oferecem suporte para a fase intermediária e final do desenvolvimento, tendo pouco ou nenhuma participação na fase inicial da concepção do sistema, onde o entendimento do foco do problema se concentra mais.

Outras características analisadas em ambientes computacionais de suporte à autoria colaborativa são:

(b) **controle de concorrência**: é necessário para evitar eventuais colisões de ações num espaço compartilhado de trabalho. Este mecanismo geralmente está baseado na resolução de colisões, ou na prevenção de concorrência. Num editor cooperativo, uma colisão ocorre quando dois ou mais usuários querem editar uma mesma seção de um documento ao mesmo tempo. Esta situação pode ser impedida através da proibição da mesma edição por dois ou mais usuários, ou através da implementação de um mecanismo de ordenação por tempo, definição de papéis, definição de prioridades, etc (MACEDO; NETO; PIMENTEL, 2000).

(c) **controle de versão**: quando um usuário estiver editando um documento, é importante que os outros usuários estejam cientes de cada nova versão

gerada. O controle de versão é importante para garantir a integridade de cada versão manipulada. Portanto, uma coordenação do trabalho se faz necessária. Dois fatores são essenciais no controle de versão: a freqüência de atualizações e a quantidade de texto a ser atualizada (MACEDO; NETO; PIMENTEL, 2000).

(d) **Paradigma de visualização**: *WYSIWIS* é o paradigma de visualização mais utilizado para visualização compartilhada. Este paradigma impõe que a freqüência de atualização de edições em documentos compartilhados seja imediata, isto é, síncrona. O grande problema deste tipo de paradigma é a falta de privacidade. Entretanto, existem outros paradigmas como o *WYSIWIMS* que permitem a edição num espaço privado e a movimentação opcional para o espaço compartilhado. O paradigma *WYSIWIMS* é explorado em editores cooperativos mistos (MACEDO; NETO; PIMENTEL, 2000).

(e) **Perspectiva dos usuários:** Segundo Ignat e Norrie (2004), um editor colaborativo deve ser uma extensão do editor mono-usuário, ou seja, com a mesma funcionalidade e característica, com adicional de suporte para grupo colaborativo, visto que os editores mono-usuários são bem aceitos. As complexidades que envolvem as aplicações colaborativas devem estar ocultas para o usuário final. Operações como criar um objeto, mover, apagar etc. devem ser feitas normalmente, cabendo ao suporte de colaboração da ferramenta o encargo do cuidar das eventuais ocorrências de inconsistência.

3.4. EXEMPLOS DE EDITORES COLABORATIVOS

Os editores colaborativos são classificados em três categorias: síncronos, assíncronos e mistos (MACEDO; NETO; PIMENTEL, 2000).

Os editores colaborativos síncronos devem oferecer mecanismos eficientes de controle de concorrência e de atualização automática. Esses editores devem possibilitar a edição simultânea de seções

de um mesmo texto. Por exemplo, a um usuário, deve ser permitida a edição de uma frase do texto, enquanto que a outro, a atualização da frase seguinte. Todavia, ambos devem visualizar, ao mesmo tempo, o que o outro está fazendo para que não ocorram conflitos, nem inconsistências.

Existe outro tipo de editor colaborativo síncrono com um usuário-editor e vários usuários-revisores/corretores. São exemplos de editores colaborativos síncronos:

- **GROVE**: é um editor multi-usuário síncrono para a criação de documentos a partir do seu esboço (Ellis et al., 1991). A estruturação preliminar é utilizada para a organização do documento e para o controle de acesso concorrente. O GROVE foi projetado para ser utilizado em encontros do tipo face-a-face ou remotos síncronos. Para cada usuário existe uma visão diferente dependendo do número de nós abertos no documento. Não há definição explícita de papéis.

- ***MACE***: é um editor colaborativo síncrono que utiliza como controle de concorrência o bloqueio de partes do documento, segundo a definição de papéis Newman-Wolfe & Pelimuhandiram (1991). MACE utiliza o paradigma de visualização WYSIWIMS através do qual o emissor (editor) e os receptores (visualizadores) têm controle sobre a granuloridade da visão compartilhada.

Os editores colaborativos assíncronos são ambientes que suportam a edição, em tempos distintos, de um documento compartilhado. São exemplos de editores colaborativos assíncronos: (MACEDO; NETO; PIMENTEL, 2000)

- ***PREP*** (*Work in Preparation*): é um ambiente assíncrono de edição colaborativa com suporte à autoria, revisão e inserção de comentários durante o processo de edição (Neuwirth et al., 1990). Este ambiente enfatiza o aspecto de comunicação através da representação visual da informação. No Prep não há definição de papéis explicitamente, entretanto, existe a definição de níveis de permissão de atualizações.

- ***QUILT***: é um sistema para autoria colaborativa assíncrona que gera uma infra-estrutura de gerenciamento de aspectos de cooperação do grupo de autores com distribuição de papéis (Dourish & Bellotti, 1992). Durante a cooperação, o Quilt adota, basicamente, enfoque hipermídia para anotação de texto, mecanismos de *e-mail* e conferência eletrônica para discussão entre os usuários-editores. Os direitos de acesso são atribuídos a cada editor de acordo com o seu papel, definido explicitamente a partir da natureza da informação e do estágio em que o trabalho se encontra.

Os Editores colaborativos mistos são aqueles que podem ser utilizados para edição cooperativa de documentos, tanto síncrona, quanto assincronamente. São exemplos de editores colaborativos mistos (MACEDO; NETO; PIMENTEL, 2000):

- ***SASSE***: implementa a metáfora de espaço compartilhado de trabalho através do documento-texto que é visualizado num quadro (*frame*) síncrono e comum a todos os componentes do grupo de trabalho (Baecker et al., 1994). Este *frame* garante a visualização *WYSIWIS* do documento. As pessoas podem trabalhar numa mesma seção, ou em seções distintas, quando editando simultaneamente. Para o caso de edição numa mesma seção, existe uma figura em miniatura da seção em questão que realça através de diversas cores as várias frases que estão sendo editadas pelos usuários num determinado momento.

- ***SEPIA***: possui basicamente, dois tipos de autoria: (1) usuários-editores trabalham, simultaneamente (síncrono) ou em tempos diferentes (assíncrono) sobre partes distintas da base de informação e (2) usuários editores trabalham sincronamente na mesma porção de informação através do ambiente de conferência que permite a cooperação-coordenação das atividades (Streitz, 1993).

> "Os principais pontos de investigação para sistemas de edição cooperativa ocorrem no contexto de edição cooperativa síncrona. Este tipo de cooperação demanda mecanismos mais

> sofisticados de controle de concorrência/consistência, visualização e controle de versão (MACEDO et al., 2000)".

3.5. EXEMPLO DE UMA FERRAMENTA *CASE* COLABORATIVA

A ferramenta *CASE Knigth* incorpora todas as funcionalidades de uma ferramenta *CASE* convencional. Utiliza um quadro branco eletrônico de grandes proporções (telão) para auxiliar a cooperação dos diversos engenheiros, desenvolvedores de software, mais a mistura de elementos (figuras) formais e informais no auxílio da resolução do problema.

4. UML

A importância de um modelo tem se verificado em todas as áreas da engenharia que conduz a representação do que se pretende ou do que se quer construir. Nas engenharias tradicionais, objetos (coisas), tais como uma máquina, casa, prédio, ou mesmo algo abstrato são representados por desenhos, figuras que auxiliam na especificação de como se quer executar o que foi previamente concebido (HARMON; WATSON, 1998).

Os desenhos e figuras são modelos de alguma coisa. Um modelo é uma descrição de algo. O trabalho de construção do modelo é chamado de modelagem, que o desenvolvedor ou projetista utiliza para investigar os requisitos necessários para que seus projetos (casa, máquina, produto) possam ser executados com sucesso.

Em se tratando de sistemas, os modelos com suas formas geométricas, desenhos e conectores são utilizados como descritores visuais de como os sistemas são ou serão operacionalizados. A modelagem é muito relevante para a representação dos sistemas, proporcionando um melhor entendimento e sincronizando a comunicação entre usuários, clientes e desenvolvedores.

Ao longo do tempo, muitos modelos e notações surgiram para que os objetivos anteriormente citados fossem atingidos. O número grande de modelos disponíveis para se utilizar, trouxe uma certa confusão e competição, principalmente entre os desenvolvedores de OO (Oriented-Object), de qual modelo seria o “melhor” ou qual o mais sofisticado no emprego da modelagem.

Essa situação levou *Grady Booch, James Rumbaugh* e Ivar Jacobson, dentre outros pesquisadores, a proporem a unificação de vários modelos. Um dos resultados dessa unificação foi o surgimento da UML.

A UML foi lançada para estabelecer a padronização de uma notação que fosse suficiente para suprir quaisquer requisitos da área de engenharia de software (HARMON; WATSON, 1998).

4.1. ORIGEM

Com a popularização das linguagens orientadas a objetos, com linguagens como C++ e *Smalltalk* na década de 80, alguns métodos de OO tornaram-se bastante populares na década de 90. Dentre esses pode-se citar:

- *Booch*: o método de *Booch*, que desenvolveu um método OO na qual definiu a notação de que um sistema é analisado com um número de visões, onde cada visão é descrita por um modelo de diagrama;

- *OMT:* é um método que foi desenvolvido na *General Eletric* onde *James Rumbaugh* trabalhou. Nese método, os sistemas são descritos por um número de modelos: o modelo de objetos, o modelo dinâmico e o modelo funcional

- *OOSE/Objectory*: o método OOSE foi construído com base no ponto de vista formulado por *Ivar Jacobson*;

- Fusion: veio da *Hewllet-Packard (D. Coleman)*, e é chamado o método da segunda geração, porque é baseado nas experiências de métodos anteriores. Inclui a técnica de especificação de operações e interações entre os objetos.

- *Coad/Yordon*: é conhecido como *OOA/OOD*. Foi um dos primeiros métodos usados em análise e desenho para OO, e é um método simples e fácil de entender.

Cada um desses métodos possui sua própria notação (seus próprios símbolos para representar modelos orientados a objetos), processos (que atividades são desenvolvidas em diferentes partes do desenvolvimento), e

ferramentas (as ferramentas CASE que suportam cada uma destas notações e processos) (HARMON; WATSON, 1998).

4.2. OS OBJETIVOS DA UML

Os objetivos da UML são:

- a modelagem de sistemas (não apenas de software) usando os conceitos da orientação a objetos;
- estabelecer uma união fazendo com que métodos conceituais sejam também executáveis;
- criar uma linguagem de modelagem utilizável tanto pelo homem quanto pela máquina. A UML está destinada a ser dominante, a linguagem de modelagem comum a ser usada nas indústrias. Está totalmente baseada em conceitos e padrões extensivamente testados que são provenientes das metodologias existentes anteriormente, e também é muito bem documentada, com toda a especificação da semântica da linguagem representada em meta-modelos.

4.3. UTILIZAÇÃO DA UML

A UML destina-se, principalmente, a sistemas complexos de software. Porém, tem sido empregada de maneira efetiva em domínios, como os seguintes:

- sistemas de informações corporativos;
- serviços bancários e financeiros;
- telecomunicações;
- transportes;
- defesa/espaço aéreo;

- vendas de varejo;
- eletrônica médica;
- científicos; e
- serviços distribuídos na Web.

A UML não está restrita à modelagem de software, podendo ser empregada com eficiência para modelar outros sistemas que não sejam de software, como fluxo de trabalho no sistema legal, a estrutura e o comportamento de sistemas e o projeto de hardware (BOOCH; JACOBSON; RUMBAUGH, 2000).

O objetivo da UML é descrever qualquer tipo de sistema, em termos de diagramas orientados a objetos. Naturalmente, o uso mais comum é para criar modelos de sistemas de software, mas a UML também é usada para representar sistemas mecânicos sem nenhum software. Seguem alguns tipos diferentes de sistemas com suas características mais comuns: (1) Sistemas de Informação; (2) sistemas Técnicos; (3) Sistemas Real-time Integrados; (4) Sistemas Distribuídos; (5) Sistemas de Software; e (6) Sistemas de Negócios (HARMON; WATSON, 1998).

4.4. SIMBOLOGIA DA UML

A simbologia utilizada pela UML divide-se em categorias: (1) núcleo ; (2) semi extensão especializada, também chamado de ornamental; e (3) os super especializado chamados de estereótipos. Os símbolos de primeira classe são divididos em dois grupos: (1) símbolos de modelagem; e (2) símbolos de relacionamento, como ilustra a Figura 11 (HARMON; WATSON, 1998):

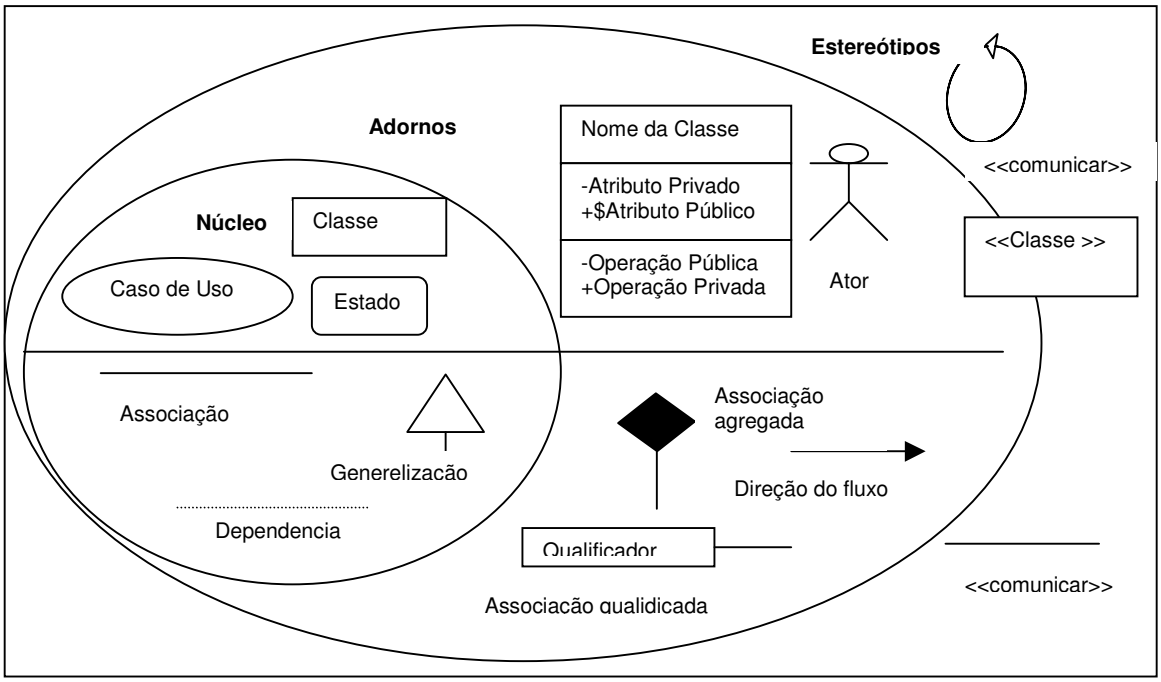


FIGURA 11 - NOTAÇÃO DA UML (HARMON; WATSON, 1998)

4.5. DIAGRAMAS DA UML

São cinco grupos diferentes de diagramas apresentados pela UML no auxílio de análise de um mesmo problema, possibilitando cinco tipos de perspectivas diferentes. São agrupados em (Harmon e Watson, 1998):

- Diagrama de Caso de Uso
- Descrição de Caso de Uso
- (OOSE) Modelo de Objeto Ideal
- Diagrama de Estrutura Estática
- Diagrama de Classe
- Diagrama de Objeto
- Diagrama de Interação
- Diagrama de Seqüência
- Diagrama de Colaboração

- Diagrama de Estado
- Diagrama de Atividade
- Diagrama de Implementação
- Diagrama de Pacotes
- Diagrama de Componentes

> "Um diagrama de classe estático descreve a estrutura estática (tipos de objetos e seus relacionamentos) de um sistema. Em outras palavras, ele descreve como o sistema está estruturado e não como ele se comporta (Eriksson et al., 1998 & Lee et al., 2001). Descreve "o que são coisas" e seus relacionamentos estáticos com outras coisas" (LEE; TEPFENHART, 2001).

Um dos propósitos do diagrama de classe é definir uma fundamentação para outros diagramas (Objetos, colaboração, dinâmico) onde outros aspectos do sistema são mostrados. A classe é um diagrama que pode ser implementado diretamente numa linguagem de programação OO (ERIKSSON; PENKER, 1998).

> "Classes definem os tipos de objetos que existem dentro de um sistema. Classes podem ter atributos que são geralmente membros de dados primitivos de objetos e operações definidoras de métodos que podem ser aplicados sobre os objetos. A visibilidade de atributos e operações pode ser definida, igualmente suas assinaturas, incluindo tipos, valores padrão, parâmetros, tipos de parâmetros e tipos de retorno" (LEE; TEPFENHART, 2001).

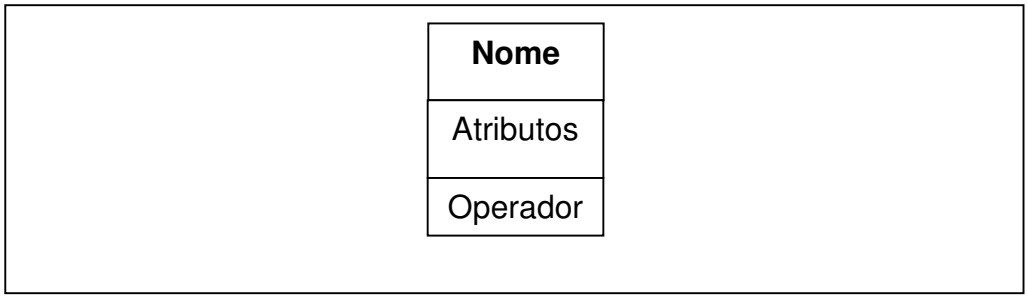


FIGURA 12 - UMA CLASSE CONFORME ESPECIFICAÇÃO DA UML (ERIKSSON; PENKER, 1998)

Uma classe é representada por um retângulo, dividido em três compartimentos: (1) o nome da classe; (2) os atributos da classe ; (3) e os operadores desta classe, como mostrado na figura 12 (ERIKSSON; PENKER, 1998).

4.6. REPRESENTAÇÃO DE CLASSES

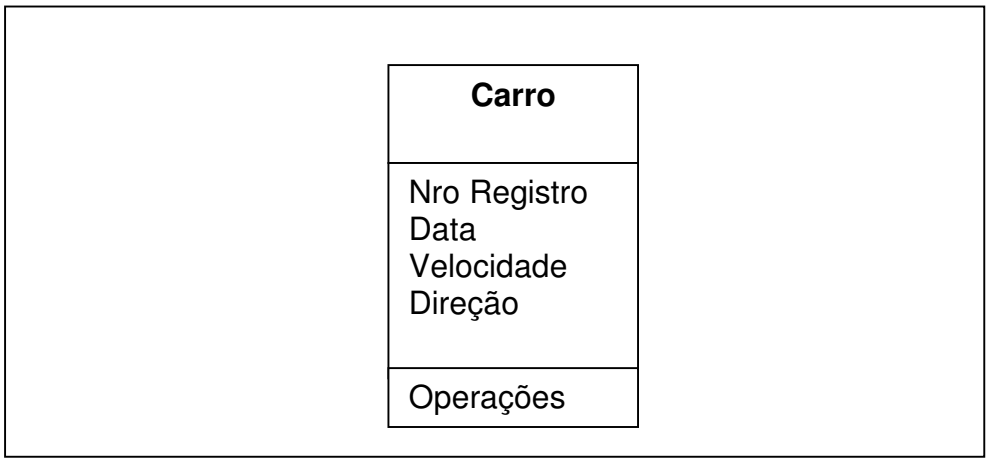


FIGURA 13 - UMA CLASSE CONFORME ESPECIFICAÇÃO DA UML (ERIKSSON; PENKER, 1998)

O nome da classe deve estar no retângulo superior e ser centralizado em negrito. À medida do possível o seu nome deve ser sugestivo para com o domínio do problema que está sendo analisado e não ambíguo, como no exemplo da figura 13.

As classes possuem atributos que descrevem as características dos objetos, conforme ilustra a Figura 13, tais como: número do registro, data, velocidade e direção. A correta inclusão de um atributo da classe mostra a informação que descreve e identifica uma instância específica dessa classe. Entretanto, somente os atributos que são de importância relevante dentro de um sistema é que deverão ser eleitos para serem modelados. Os atributos podem ser: Inteiro, Lógico, Real (atributos primários) ou qualquer outro tipo pode ser usado, incluindo outras classes (ERIKSSON; PENKER, 1998).

A notação básica da UML, para representar atributos são (HARMON et al., 1998 & LEE et al., 2001):

Visibilidade atributo: tipo de dados = valor *default* (padrão)

- avisibilidade pode ser: (+) pública; (#) protegida; ou (-) privada;
- nome é uma cadeia de caracteres (*string*) pela qual o atributo é identificado;
- tipo de dados é o tipo do atributo: Inteiro, Real, Lógico, etc;
- valor padrão é um valor designado ao atributo inicialmente.

As operações de uma classe são representadas de acordo com a seguinte notação:

Visibilidade operação (lista de argumentos):tipo do retorno

- A visibilidade pode ser: (+) pública; (#) protegida; ou (-) privada;
- Nome é uma cadeia de caracteres (string) pela qual a operação é identificada;
- Lista de argumentos (parâmetros) contém parâmetros separados por vírgula, os quais são dados por:
    - *direção nome: tipo = padrão;*
    - *direção* indica se o parâmetro é para *input (in), output (out)* ou para ambos *(inout);*
    - *Nome*: é o nome do parâmetro;
    - *tipo*: é o tipo do parâmetro;
    - *valor padrão*: identifica o valor padrão (inicial) do parâmetro.

4.7. ASSOCIAÇÃO

É uma relação que permite especificar quais objetos de uma dada classe relacionam-se com objetos de outra classe. As associações podem ter um nome, que por sua vez pode ter uma seta associada, indicando a direção em que o nome deve ser lido (LEE; TEPFENHART, 2001).

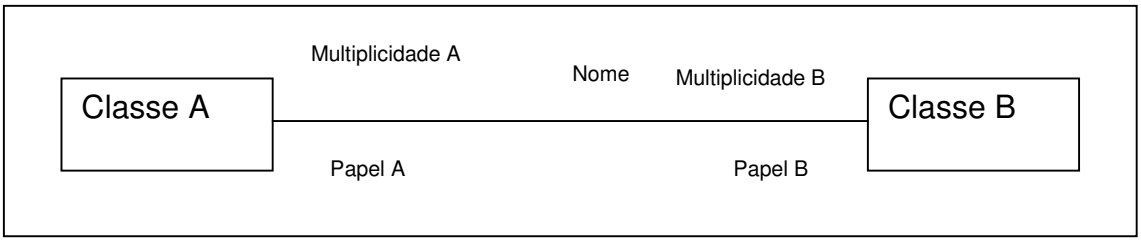


FIGURA 14 - ASSOCIAÇÃO BÁSICA DA UML ENTRE CLASSES (LEE, TEPFENHART, 2001)

A figura 14 mostra a representação básica de uma associação na UML. Uma associação é ilustrada como uma linha conectando duas classes. A linha é rotulada com o nome da associação, das multiplicidades das duas classes (número de objetos da determinada classe) participando da associação e os papéis que as instâncias de cada classe assumem dentro da associação. Todos esses rótulos são opcionais.

4.8. A MULTIPLICIDADE DE UMA ASSOCIAÇÃO

A multiplicidade associada a uma extremidade de uma classe pode ser dada da seguinte forma:

Um para um

Um para muitos

Um para um ou mais

Um para zero ou um

Um para exato

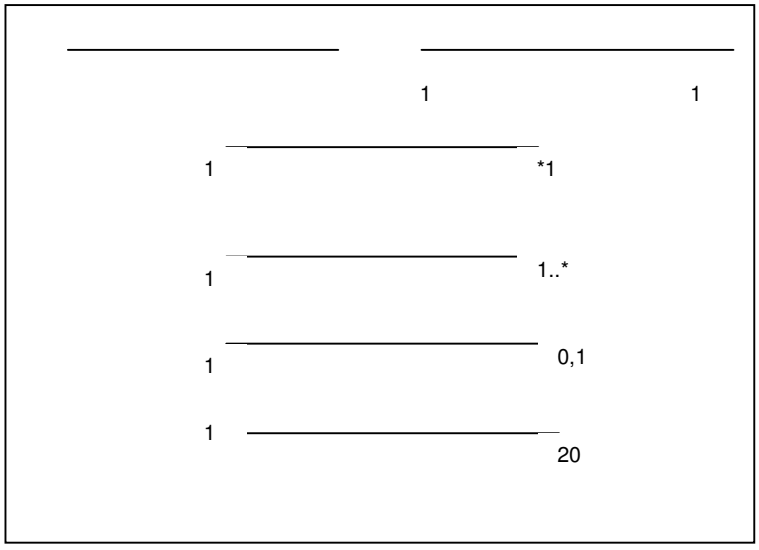


FIGURA 15 - DIREÇÃO DA MULTIPLICIDADE DA ASSOCIAÇÃO

A direção das associações é bidirecional quando não se especifica a linha com uma seta. No caso em que a direção é única, indica-se a direção da seta, como na figura abaixo:

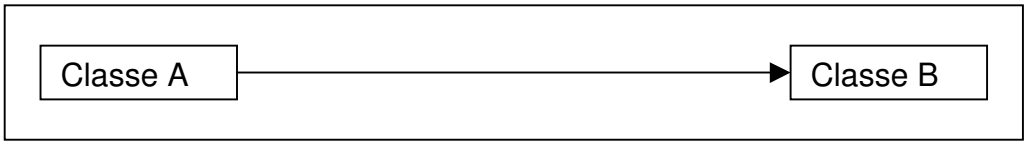


FIGURA 16 - ASSOCIAÇÃO ENTRE CLASSES UNILATERAL

4.9. ASSOCIAÇÃO DERIVADA

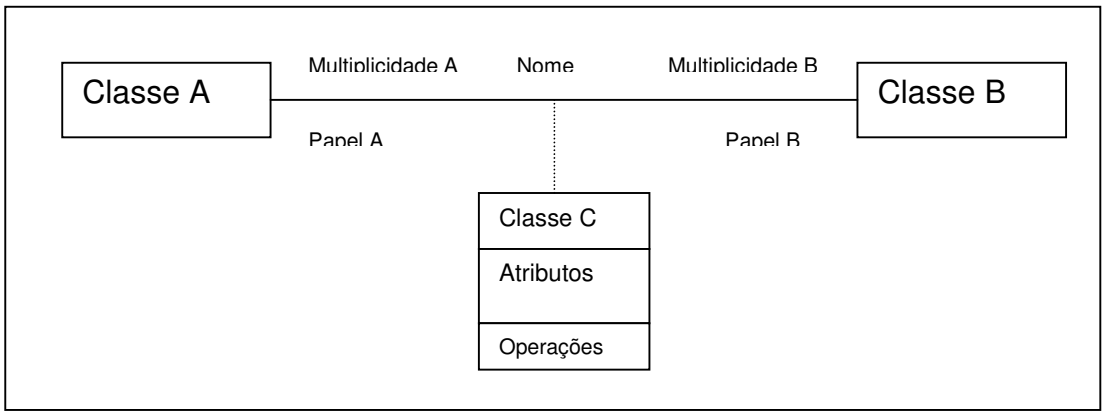


FIGURA 17 - ASSOCIAÇÃO DE CLASSES

A classe derivada (classe de associação – Classe C) representada na figura 17, é uma conseqüência da associação entre duas outras classes (A e B). Também chamado de relacionamento dependente, essa classe pode ter atributos e operações associados. (LEE; TEPFENHART, 2001).

A dependência indica que se um dos elementos muda, então o outro também mudará.

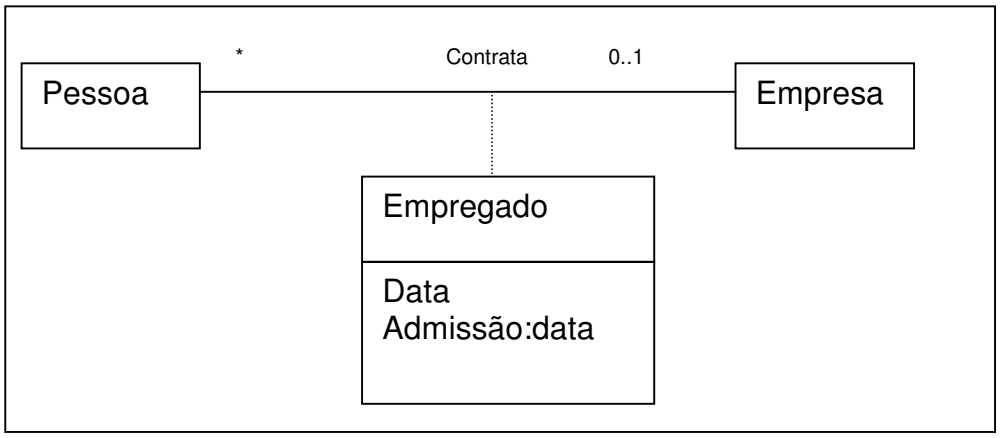


FIGURA 18 - ASSOCIAÇÃO DE CLASSES

Conforme mostrado na figura 18, que representa uma modelagem entre duas classes (Pessoa e Empresa). Da associação dessas duas classes surge a classe empregado, que necessita de um atributo que não pertence a nenhuma das duas classes (Pessoa e Empresa), a data da admissão do empregado.

4.10. ASSOCIAÇÃO QUALIFICADA

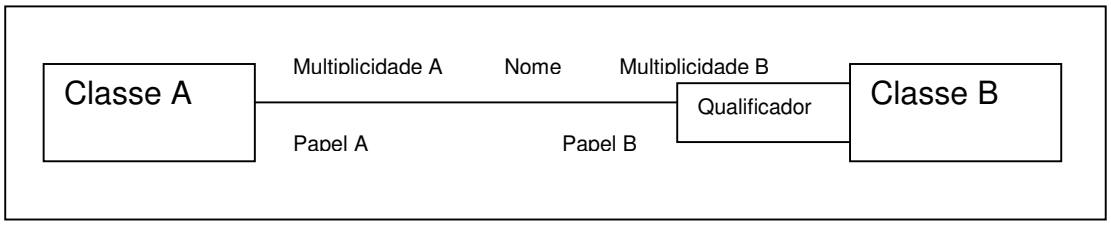


FIGURA 19 - ASSOCIAÇÃO ENTRE CLASSES UNILATERAL

O Qualificador é especificado quando se quer determinar mais precisamente a natureza da associação. Qual o atributo da Classe B será utilizado na associação com a classe A? Ou seja, se há um desejo de se qualificar a associação, deixar explícito qual o atributo do relacionamento. Pode- se dizer também, que todo acesso que a classe A fizer na classe B, o atributo qualificador, filtro, deverá ser passado como argumento (BOOCH; JACOBSON; RUMBAUGH, 1997).

4.11. ASSOCIAÇÃO RECURSIVA

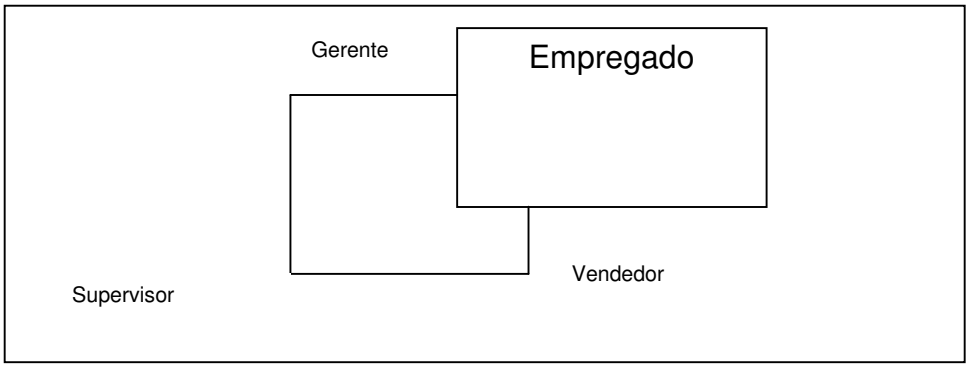


FIGURA 20 - NOTAÇÃO PARA UMA ASSOCIAÇÃO RECURSIVA

Uma associação recursiva acontece quando uma classe se relaciona consigo mesma. O uso de papéis (Gerente, Supervisor, Vendedor) é utilizado para clarear o entendimento desse tipo de relacionamento. Na figura 20, alguns objetos da classe Empregado são Gerentes, outros são supervisores e vendedores. Os relacionamentos entre eles dizem que um

gerente gerencia vários supervisores que, por sua vez, supervisionam vários vendedores.

4.12. AGREGAÇÃO E COMPOSIÇÃO

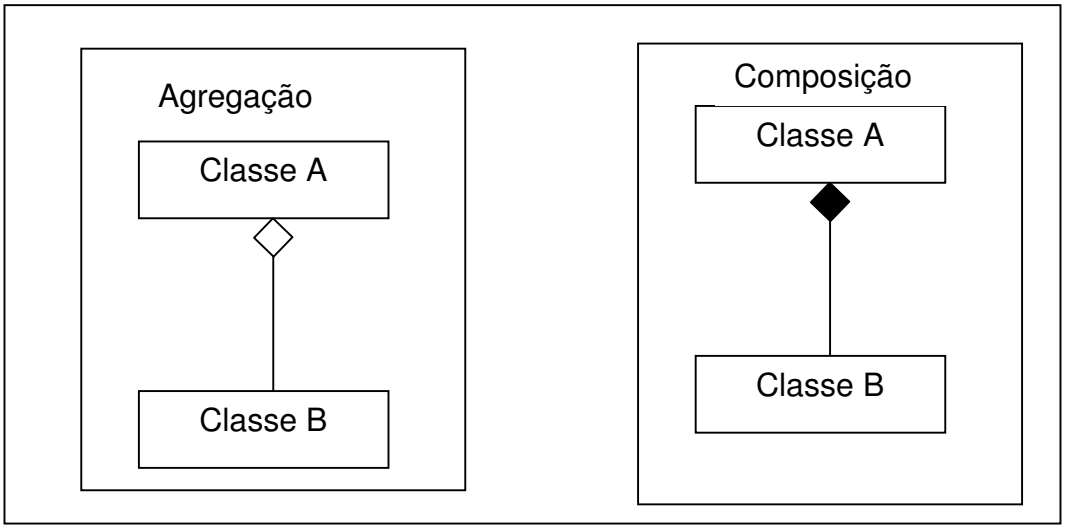


FIGURA 21 – NOTAÇÃO PARA AGREGAÇÃO E COMPOSIÇÃO

Especifica uma relação *"todo – parte", que r*epresenta uma relação mais forte que uma associação. As palavras-chaves utilizadas para identificar uma agregação são:

- "consiste em..."
- "contém..."
- "é parte de..."

A Composição é uma forma de agregação na qual a relação mais forte está entre o objeto agregador e os objetos componentes:

- cada componente pertence a um único conjunto
- um componente só faz sentido na composição
- quando desaparece a composição o componente é também destruído

4.13. GENERALIZAÇÃO E ESPECIALIZAÇÃO

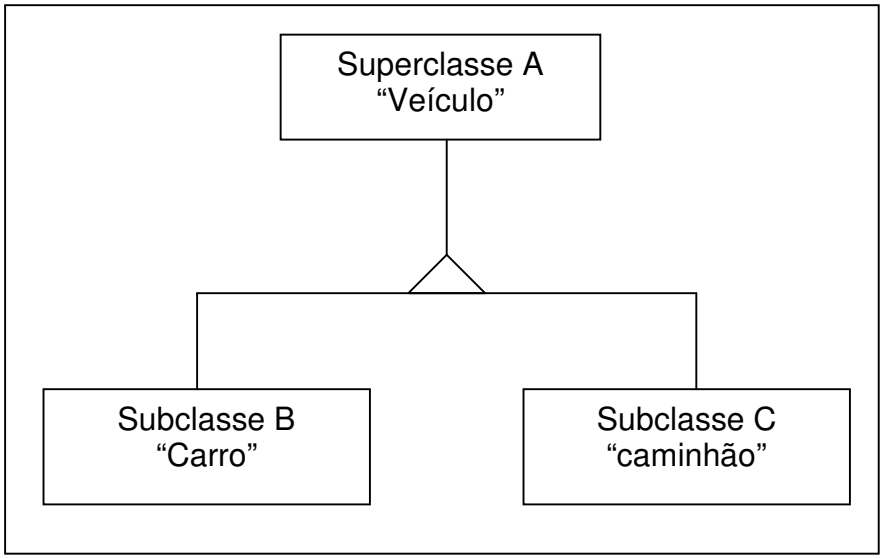


FIGURA 22 - REPRESENTAÇÃO DA GENERALIZAÇÃO E ESPECIALIZAÇÃO

A Generalização é um relacionamento entre uma classe genérica e uma específica, onde a classe específica é chamada de Subclasse, que herda todas as características da classe mais geral, Superclasse (BOOCH; JACOBSON; RUMBAUGH, 2000).

O elemento específico *(subclass*e B e C) contém mais informação que o genérico *(superclass*e A*),* mas é totalmente consistente com o mesmo. As subclasses herdam os atributos, operações e relações da superclasse. Especifica uma relação *“*é”. Por exemplo, “Um carro é um veículo” e “Um caminhão é um veículo”.

5. POLÍTICA DE COLABORAÇÃO

Uma política de colaboração baseia-se em um conjunto de regras, diretrizes, e mesmo ações que devem ser observadas para que um ambiente de colaborativo possa ser governado.

Segundo Yokota (1999) e Edwards (1996), todo ambiente tem a necessidade de ter uma política de colaboração, para que não se torne caótico, apesar da sua complexidade. As regras dessa política devem definir objeções, considerações e procedimentos que envolvam os elementos (atores, ações, objetos, tempo e espaço (CAMOLESI, 2005)) do ambiente colaborativo, bem como ter a preocupação com a segurança e o controle de acesso. Para Edwards (1996), qualquer mecanismo que ofereça a redução ou mesmo a eliminação de complexidades e conflitos, é muito bem-vindo, pois tal como no mundo real o ambiente colaborativo é dinâmico e o aspecto surpresa está sempre presente. Contudo, uma política deve manter a riqueza das interações e fluidez, com estabilidade do ambiente, não sendo, então, um processo inibidor.

Um dos componentes vitais para quaisquer aplicações ou ambientes colaborativos é o aspecto relativo à segurança e ao controle de acesso. Porém, a segurança é vista como um fator complicador na implementação (EDWARDS, 1996).

Em um ambiente complexo, como o dos ambientes colaborativos, qualquer mecanismo que ofereça a redução, ou mesmo a eliminação dessas complexidades e conflitos, é muito bem-vindo (EDWARDS, 1996).

Há três requisitos básicos para um modelo de acesso colaborativo (BULLOK; BENFORD, 1999):

- o mecanismo deve ser simples;

- o mecanismo deve ser moderado e discreto para o usuário, sem que haja sobre os mesmos uma sobrecarga, assegurando de forma confiável as operações. Isso significa que o mecanismo deve ser naturalmente integrado dentro da filosofia do sistema;

- deve ser fácil para inspeção e mudança dos direitos de acesso.

Sobretudo, os efeitos de controles de acessos devem ser entendidos e as conseqüências de qualquer mudança devem estar bem claras.

Tem sido reconhecida a importância da segurança e do controle de acesso. Poucos trabalhos foram publicados desde 1992, quando SHEN e DEWAN publicaram trabalho sobre o assunto. Porém, recentemente, essa tendência mudou e mais modelos têm sido propostos (BULLOK; BENFORD, 1999).

Como mais e mais pessoas passam a trabalhar juntas em forma de equipes colaborativas e formação de grupos de trabalho em diversas áreas de trabalho, acentua-se a importância de fornecer um mecanismo de acesso simples, flexível e que dê suporte a esses grupos tornando seus trabalhos seguros e confiáveis (BULLOK; BENFORD, 1999; XU, 2005).

Quando se fala em sistemas, principalmente em sistemas de computadores, logo vem uma preocupação quanto à segurança em seus vários aspectos e formas. Mas essa preocupação é mais crítica, principalmente no que diz respeito a controle de acesso e na capacidade desse sistema controlar todo o tipo de alteração (EDWARDS, 1996).

Quando se trata de sistemas colaborativos, essa preocupação é ainda maior, pois esses ambientes são potencialmente caóticos. Múltiplos usuários estão presentes colaborando na criação e execução de tarefas e as interações podem ocorrer de forma imprevisível e inesperada (EDWARDS, 1996).

Em sistemas mono-usuários todas as ações e interações são perfeitamente previsíveis, pois sempre há um único usuário no contexto do

sistema. A simples inserção de outro usuário no contexto do ambiente, torna-o colaborativo, e, a partir dessa mudança, introduz-se também um ambiente de incertezas e ações imprevisíveis, com o aspecto surpresa sempre presente (EDWARDS, 1996).

A complexidade dos ambientes colaborativos é muito alta, pois o objetivo é proporcionar o mesmo dinamismo presente no mundo real. Nessa abordagem, torna-se possível que os usuários tenham inúmeras novas interações que lhes proporcionem riqueza e fluidez nos seus objetivos (EDWARDS, 1996).

Mas, esses ambientes têm que dar conta das distrações e demandas crescentes dos usuários, dentro de limites toleráveis, para manter esse ambiente colaborativo sustentável (EDWARDS, 1996).

No intuito de ter um controle mais efetivo do ambiente colaborativo, alguns trabalhos realizados implementam: (1) o conceito de sessão de gerenciamento (convida um usuário a entrar na sessão ou paginar uma lista de sessões); (2) criação e manutenção de regras dos usuários no editor da colaboração; (3) aceite ou rejeição de pedidos de acesso à sessão e a dados (FARIAS, 2002 & EDWARDS, 1996).

A preocupação de controle e a decisão do que fazer com as ações inesperadas dos usuários, que ocorrem nos ambientes colaborativos, devem ser de responsabilidade da aplicação e não dos usuários “mestres” da sessão, ou seja a decisão se determinadas ações devem provocar a exclusão ou não do usuário nocivo, ou limitações de suas ações (EDWARDS, 1996).

As aplicações devem ser providas de diretrizes para dar suporte ao dinamismo dos ambientes colaborativos e deixar que os usuários exerçam o seu papel de colaborador, em vez de coordenador desses novos usuários (EDWARDS, 1996).

Para responder as essas ocorrências e evitar o caos em ambientes colaborativos, introduz-se o conceito de política de colaboração. Essa política governará como os usuários e aplicações e irão interagir uns com

os outros, como ocorre no mundo real. Essa política descreverá em linhas gerais o procedimento em cada evento dinamicamente ocorrido (EDWARDS, 1996).

Os principais objetivos de uma política de colaboração devem ser:

- reduzir a imprevisibilidade no sistema, dando níveis de gerenciamento para permitir responder às ações de usuários e outras aplicações e suportar o princípio de "menor surpresa" ("l*east surprise*")

- requerer o mínimo de esforço dos usuários, ou seja, mantê-los livres para usarem o sistema em vez de preocuparem em administrá-lo.

É preciso, por um lado, não impor restrições, por outro não inibir as interações, ou seja, estimular a colaboração.

Em essência, uma política de colaboração é usada para limitar seletivamente o dinamismo pertinente em sistemas colaborativos e, por outro lado, manter a riqueza proporcionada pelas inúmeras interações de forma gerenciável, pois essas interações são componentes indispensáveis na comunicação entre usuários (EDWARDS, 1996).

Uma das implementações de suporte à política de colaboração é o uso do controle de acesso. Esse controle é provido de uma ferramenta poderosa que é capaz de capturar muitas das situações ocorridas dentro do ambiente colaborativo. Esse suporte é tanto no aspecto de política estática, como também, principalmente, no aspecto dinâmico da colaboração (EDWARDS, 1996).

Muitas das situações que ocorrem em trabalhos colaborativos usados no mundo real é comum a todas as linguagens humanas. Se forem capturados e receberem suporte do sistema de controle de acesso, os ambientes colaborativos computadorizados tornam-se muito próximos do praticado em ambientes convencionais, como por exemplo (EDWARDS, 1996):

"*Eu não me importo se determinadas pessoas então conhecendo o que estou fazendo, mas outras pessoas*";

"*Não quero ser incomodado por ninguém quando estou trabalhando, a menos que seja o meu chefe*";

"*Eu quero compartilhar meu espaço de trabalho com outras pessoas durante determinada atividade...*";

"*Se alguém me chamar, diga que não estou, a menos que ...*".

Esses são alguns exemplos de cenários que ocorrem no mundo real e que podem ser capturados e definidos em termos de direito de controle de acesso para permitir que o sistema responda a cada uma das eventualidades (EDWARDS, 1996).

A política de colaboração pode ser dividida em áreas, tais como:

- **política de domínio de coordenação**, que preocupa-se com o controle entre aplicações e informações, e sua localização.

- **política de domínio das aplicações**, que preocupam-se em controlar os dados de uma aplicação específica, tal como o acesso a um texto de um editor compartilhado.

Enquanto a política de domínio de coordenação é um controle que tende a ser tudo ou nada, ou seja, uma política de acesso mais rígida, a política de domínio da aplicação introduz o conceito de papéis para definir o controle de acesso de grupos. Um papel é uma categoria de usuário, dentro de uma população de usuários de uma determinada aplicação, onde todos esses usuários herdam um conjunto de direitos de controle de acesso dentro da aplicação. Esse tipo de categoria é largamente utilizada em editores compartilhados como o *Quilt*, que inclui papéis para escrita, leituras e comentários. Papéis também são encontrados em várias outras aplicações colaborativas (EDWARDS, 1996).

A aplicação da política de colaboração em termos de aplicações como papéis, pode se dar de duas formas : como papéis estáticos e dinâmicos, conforme descrito a seguir (EDWARDS, 1996).

5.1. PAPÉIS ESTÁTICOS

Em uso tradicional, papéis são tipicamente estáticos, pois as regras que são definidas raramente mudam, e, um papel é definido em termos dos usuários que são membros desses mesmos papéis (EDWARDS, 1996).

Outra característica importante é que o uso de papéis tem poucas características em comum entre os diversos sistemas, pois (EDWARDS, 1996):

- o conjunto de papéis é determinado a *priori* pelo ambiente e pela aplicação;
- os integrantes, usuários desses papéis, são precisamente determinados no início do ciclo de vida de uma sessão;
- os papéis são específicos em termos de usuário;
- há pouca ou quase nenhuma mudança para as regras dos usuários durante uma sessão.

Apesar do largo uso de papéis estáticos, visto que se pode definir a maioria das ações dos membros de uma sessão, e, isso é feito no início de uma sessão, a flexibilidade fica prejudicada e qualquer variação do ambiente e situação que ocorrer e que não estão previstas, são transferidas para o usuário, que decide o que fazer. Isso acarreta uma carga muito pesada para o usuário (EDWARDS, 1996).

Uma das vantagens que se pode citar do uso de papéis estáticos é a fácil implementação, pois são bem conhecidos pelos desenvolvedores e, em muitas situações são suficientes em termos de modelo de política de direito de acesso (EDWARDS, 1996).

As desvantagens são:

- exige definição explícita dos usuários em usar tais papéis;
- ignora as situações dinâmicas, como ocorre no mundo real;
- as especificações são rígidas e não flexíveis;
- os papéis são definidos em termos de usuários ou do ambiente.

5.2. PAPÉIS DINÂMICOS

Superior ao modelo estático, o modelo de papéis dinâmicos é muito flexível, embora seja mais caro e mais difícil de gerenciamento. São caracterizadas pela descrição em termos de seus atributos, ao invés de definição (fixação) dos papéis, como no modelo estático.

Segundo Edwards (1996), esse modelo é caracterizado pela definição de papéis em tempo de uso do sistema, à medida em que as requisições são feitas.

Algumas propriedades inerentes ao modelo de papéis dinâmicos são:

- permitem que os papéis dos grupos de usuários sejam baseados em atributos, ao invés da atribuição do *user name*, como no modelo estático;
- os direitos dos usuários dos grupos podem variar, tornando-se mais, ou menos poderosos durante o ciclo de vida de uma sessão;
- o direito de acesso pode ser garantido em determinada circunstância.

Pelo uso de descrição de papéis nos grupos de usuários, do que a definição (estática), os usuários podem ficar mais aliviados de algumas cargas indesejáveis, como de dar manutenção aos papéis do grupo de usuário explicitamente.

Um bom exemplo da diferença do uso de papéis dinâmicos, ao invés dos estáticos, é poder não somente especificar um papel da forma: "*pessoas que compartilham minha área de trabalho",* mas, "*pessoas que compartilham minha área de trabalho* ***no momento****"* (EDWARDS, 1996).

Além disso, a aplicação de papéis dinâmicos traz informações sobre os aspectos dos usuários, tanto dentro do ambiente de colaboração, como da aplicação. O ambiente pode ser mais responsável quando lidar com as mudanças ocorridas durante a existência de uma sessão. Com isso, diminui consideravelmente a carga dos usuários em ter que decidir manualmente as eventuais mudanças de estado.

Há, nesse caso, uma tendência do sistema ficar mais fácil de se utilizar, pois torna-se mais "inteligente" (EDWARDS, 1996).

5.3. EXEMPLO DE IMPLEMENTAÇÃO

Segundo Edwards (1996), o *intermezzo framework* é um sistema de infra-estrutura para sistema colaborativo que fornece suporte de coordenação às aplicações, nas partes de gerenciamento de sessão, aspecto de consistência e política de controle.

Esse sistema provê mecanismo para criação de política de controle em forma de papéis, tanto na forma estática, quanto na forma dinâmica.

O *intermezzo* fornece uma linguagem de especificação de papéis que dá suporte aos dois tipos de papéis, estático e dinâmico, e, além disso, a uma terceira forma, a agregação, que é uma junção das duas formas.

A função da forma estática, que também é chamada de simples, é atribuir um papel ao *"username"*:

papel Keith = "Keith Edwards<kedwards@pare.xerox.com"

A declaração de papéis dinâmicos está ligada a uma determinada função apontando para um nome de um papel (EDWARDS, 1996):

papel demoday = [~keith/.policy/demoday.py]

A agregação de papéis provê um mecanismo de agrupamento de outros papéis, seja simples (estático), dinâmico ou mesmo outras agregações (BULLOK; BENFORD, 1999):

papéis myproj = {Keith, Beth, doug, demoday}

A ordem da aplicação dos papéis ocorre primeiramente na avaliação dos papéis estáticos e, caso não sejam obtidos resultados satisfatórios ("*true*"), é que os papéis dinâmicos são invocados (EDWARDS, 1996).

Segundo Bullok e Benford (1999), o controle de acesso também é chamado administração de acesso, que assegura que usuários não façam acesso sem autorização.

Cada controle deve dar suporte a uma proteção de fina-granulação, método flexível de atribuição de acesso, fácil implementação e de fácil uso, pelos diferentes usuários, com diferentes necessidades de proteção e oferecer um número reduzido de características (configuração) e que possam ser aprendidas de forma incremental.

Um sistema de segurança deve responder a duas perguntas básicas (Bullok e Benford, 1999):

- controle de acesso básico:

  Que tipo de controle de acesso básico um sistema de proteção deve possuir para proteger as operações sobre os objetos gerenciais?

- administração de Acesso:

  Como devem ser protegidas as suas próprias operações de especificação?

Essas perguntas têm sido exploradas intensivamente pelos sistemas de segurança tradicionais, tais como os Sistemas Operacionais e Sistema de Gerenciadores Banco de Dados. Esses sistemas oferecem a forma de lista (estáticas), as primitivas de controle de acesso sobre os objetos como arquivos e banco de dados. Oferecem também um suporte de administração de controle na definição das especificações de acesso,

Mas, com o emergir das aplicações colaborativas, esses controles ficaram aquém das necessidades inerentes ao ambientes colaborativos, que possibilitam, em tempo real a edição e composição de um documento, o desenvolvimento de software e outras funcionalidades.

Muitos dos aspectos das operações que os sistemas tradicionais oferecem são reaproveitados nas aplicações colaborativas. Porém, em primeiro lugar, não são flexíveis e as operações requerem um nível de sofisticação elevada, tais como requisitos de compartilhamento de objetos, interações entre objetos de distribuição com diferentes usuários etc. Esses objetos não são somente tabelas de arquivos relacionais, mas também planilhas, gráficos, documentos *hiper-texto* etc. Ambientes colaborativos têm uma grande variedade de natureza específica da colaboração, ou seja, são muito mais voláteis do que os sistema tradicionais.

6. A POLÍTICA DE COLABORAÇÃO PARA O AMBIENTE DE CO-AUTORIA DE DIAGRAMAS DE CLASSE.

Todo ambiente necessita de uma política, um conjunto de regras, diretrizes e procedimentos, para que não se torne caótico. A dinâmica da interatividade entre os atores envolvidos e os objetos são fatores marcantes do ambiente colaborativo e devem receber atenção para que o trabalho seja realizado de forma segura e flexível**,** eliminando os aspectos surpresas que são comuns nesse tipo de ambiente. A preocupação com o dinamismo das interações e a estabilidade do ambiente colaborativo são requisitos importantes descritos nesta proposta.

Aspectos relativos ao ambiente distribuído também fazem parte das considerações desta proposta, garantindo, assim, o controle de concorrência, a transparência quanto à localização dos atores envolvidos bem como a localização, a forma de armazenamento dos dados, entre outros aspectos. Essa proposta traz também uma possibilidade de configuração da política de colaboração, por projeto cadastrado permitindo a customização das regras e diretrizes de acordo com as características do projeto.

6.1. CONTEXTO DO AMBIENTE COLABORATIVO

O ambiente colaborativo no qual a política proposta atuará envolve os seguintes aspectos:

- **arquitetura de comunicação**: - Os aspectos de comunicação do ambiente são descritos quanto à arquitetura Cliente/Servidor*,* o tipo de comunicação, estações de trabalho e localização dos recursos funcionais (*Netware*).

- **arquitetura de infra-estrutura**: plataforma primária *Windows*, plataforma secundária *JAVA/RMI*, e interface das funcionalidades (*Taskware*) (Versey, 1995), *Internet*;

- **grupo de trabalho**: configuração de usuários e criação de papéis de usuários, que constitui uma camada social (*Teamware*), (Versey, 1995), possibilitando a organização, configuração e principalmente a execução das tarefas (HAAKE; BOURIMI, 2004);

- **funcionalidades**: as funcionalidades locais e em grupo, constituem a camada onde se dá a formação do *Groupware* (Versey, 1995), possibilitando a realização do trabalho em grupo.

6.2. ATORES ENVOLVIDOS

No ambiente colaborativo os usuários (atores) são os que constituem os agentes motivadores das interações, estimulando a comunicação, gerando resultados (CAMOLESI, 2005).

A formação de grupos organizados e com papéis bem definidos, constitui um fator determinante para a estabilidade e organização do ambiente bem como de formação da política de colaboração.

Na política de colaboração proposta foram criados três papéis de usuários, que serão descritos abaixo. Esses papéis fornecerão uma melhor administração da organização do ambiente, atribuição de responsabilidades e controle de acesso. Segundo Du Li (1999), o conceito de papéis também é adotado nos sistemas colaborativos como *INTERMEZZO*, *QUILT* e *COLA* que consideram como um excelente separador de propósito. Na política de colaboração adotada neste trabalho foram criados três papéis de usuários, que serão descritos a seguir.

6.3. COORDENADOR GERAL

Responsável pela instalação da ferramenta do lado servidor. Faz também a sua auto identificação através de uma interface de cadastramento e a criação de projetos e seus respectivos coordenadores.

6.4. COORDENADOR DE PROJETO

Responsável pela coordenação e administração do projeto que estiver sob sua responsabilidade. É de responsabilidade do coordenador formar a equipe de modeladores que atuarão no projeto sob sua responsabilidade, atribuindo direitos de acesso aos usuários modeladores, configurando a tomada de decisão para aceitação ou não dos objetos (elementos) criados no ambiente colaborativo.

6.5. MODELADOR (USUÁRIO DESENVOLVEDOR)

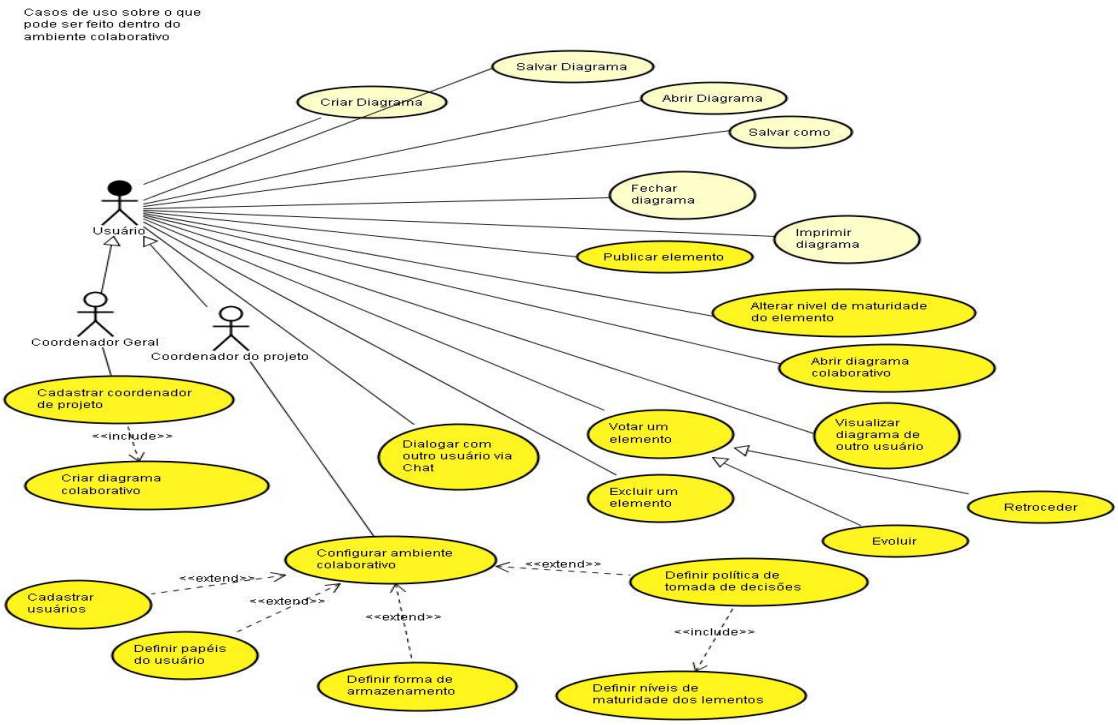


FIGURA 23 - CASOS DE USO QUE REPRESENTAM AS AÇÕES DOS USUÁRIOS NO CONTEXTO DO AMBIENTE COLABORATIVO

Designado pelo coordenador de projeto, comporá a equipe de colaboradores. Os modeladores são efetivamente os usuários que atuarão na modelagem do diagrama de classes.

Conforme ilustra a Figura 23, tem-se a modelagem de Casos de Uso que representa as ações dos usuários no ambiente colaborativo, bem

como a caracterização dos atores e seus papéis inseridos nesse ambiente. As ações que têm participação colaborativa direta estão representadas através das elipses em amarelo forte.

6.6. AMBIENTE DE TRABALHO

Há, no ambiente colaborativo, basicamente, três instâncias de ambientes de trabalho, onde a política de colaboração deve atuar, conforme descreve-se a seguir.

6.7. AMBIENTE DE TRABALHO INTERNO (MONITORAMENTO)

É um espaço de trabalho interno onde algumas ações automatizadas são realizadas pelo software. Essas ações automatizadas são as que monitoram constantemente o ambiente de trabalho colaborativo garantindo organização e sincronismo de dados e informações. Essas ações são de dois tipos:

(a) **ações repetitivas (*looping*)**: são aquelas executadas repetidas vezes enquanto o ambiente está no ar, e têm como função:

- identificar novos usuários no ambiente colaborativo e avisar os demais integrantes do grupo;
- identificar usuários que saíram do ambiente colaborativo e avisar os demais integrantes do grupo;
- reconhecer novo elemento foi publicado.

(b) **ações condicionadas**: são automatizadas, realizadas mediante o aparecimento de um evento. Essas ações provocam novas execuções de ações internas no ambiente colaborativo completando um ciclo de processamento, como por exemplo:

- ativar o ícone avisando quando há novos elementos publicados;
- ao se convocar uma tomada de decisão, deve-se inibir a ação de atualização do elemento e disponibilizar a ação de opção de voto;
- excedido o tempo de votação de um elemento, encerra-se o processo de eleição e apura-se os resultados;
- mudar a cor do elemento em todas as estações de trabalho quando esse for publicado ou alocado.

6.8. AMBIENTE DE TRABALHO INDIVIDUAL

Onde as ações dos usuários modeladores não afetam diretamente o coletivo. Esse ambiente é a área de trabalho onde se pode realizar várias ações sem prejuízo para o coletivo. Entretanto, a partir da área de trabalho individual são geradas as propostas de elementos (objetos) para a área coletiva. Essa área de trabalho serve como uma área de simulações e rascunhos. Cada modelador cadastrado num projeto tem uma área de trabalho individual.

6.9. AMBIENTE DE TRABALHO COLETIVO

É o espaço de trabalho no qual todas as ações afetam o coletivo. Cada ação tem um alto impacto para o ambiente, podendo trazer insatisfação ao trabalho em grupo, se a política de colaboração for mal definida e pouco administrada. É nesse espaço de trabalho que a política de colaboração proposta atua em maior escala, pois deve trazer um alto grau de satisfação ao participantes do grupo de trabalho.

Para as funcionalidades coletivas e para a área de trabalho coletiva, um importante aspecto da política de colaboração abordado, é a tomada de

decisão, ou seja, obter a aceitação ou a rejeição dos demais modeladores quando:

- o elemento (objeto) é públicado;
- a evolução de elemento público é requisitada. As novas atualizações no elemento precisam ser reconhecidas e aceitas pelos demais modeladores;
- o retrocesso de um elemento público é requisitado. O elemento tornou-se obsoleto e há necessidade de retroceder para possibilitar a exclusão ou a reformulação do elemento.

Assim que um objeto se torna público, ou seja, uma funcionalidade pública permitida é acionada, o ambiente coletivo é afetado, conforme mostrado na Figura 24, que ilustra os diversos estados de um objeto, desde o estado embrionário (visível apenas na área de trabalho local) até passar ao estado de público ao público consolidado.

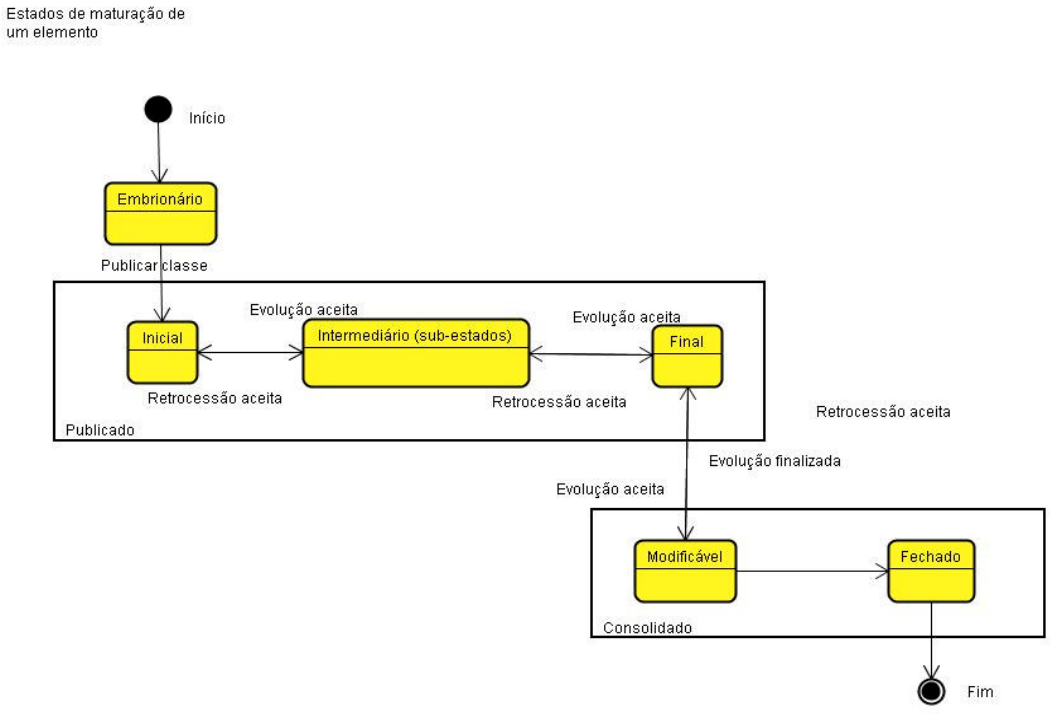


FIGURA 24 – DIAGRAMA DE ESTADOS DOS OBJETOS NO AMBIENTE COLABORATIVO

Para que as mudanças de estados, tanto evolutivas, quanto regressivas (retrocesso), sejam aceitas, uma política de decisão deve ser adotada. As seguintes formas de tomada de decisão são aceitas na política proposta:

- ***eleição simples*** – quando 50% mais um dos usuários cadastrados no projeto aceitam a mudança de estado do objeto;

- ***autoridade do coordenador*** – somente o coordenador do projeto vota aceitando ou não a mudança de estado do objeto;

- ***consenso*** – todos os usuários modeladores presentes na sessão podem votar. É considerada aceita ou não a mudança de estado do objeto quando todos os que votaram têm a mesma opinião, ou seja, todos votaram pelo *sim,* ou todos votaram pelo *não*.

A forma de tomada de decisão é um mecanismo importante dessa política de colaboração que está sendo proposta. Dependendo da complexidade do projeto e/ou da composição da equipe de modeladores, o coordenador de projeto pode estabelecer um parâmetro que melhor se adapte ao contexto do ambiente. Com esse mecanismo, o ambiente pode ser configurado em qualquer tempo, evitando conflitos. Quando os empasses são grandes entre os modeladores, o coordenador de projeto assume o controle dos elementos publicados.

É de responsabilidade do coordenador estabelecer os critérios a serem aplicados no ambiente colaborativo de seu projeto. A configuração da política de colaboração de um projeto não afeta os outros, mesmo estando em um mesmo contexto do ambiente colaborativo.

Na área de trabalho coletivo, o desenvolvedor tem possibilidade de visualizar, na estação própria de trabalho, o que cada desenvolvedor está realizando em suas respectivas áreas de trabalho. Mesmo que os elementos estejam no estado embrionário, há a possibilidade de saber

o que estão realizando. O compartilhamento quanto ao conhecimento, através do paradigma *WYSIWIS,* ou seja, que possibilita a visualização do que está passando em cada área de trabalho individual para os demais modeladores, é um requisito presente na política de colaboração proposta.

6.10. TIPOS DE FUNCIONALIDADES DO AMBIENTE COLABORATIVO

Também para as funcionalidades, conforme ilustra a Figura 23, pode-se observar aquelas que são de uso individual (elipses em amarelo claro) e coletiva (São as elipses em amarelo escuro), tais como:

6.11. FUNCIONALIDADES LOCAIS E INDIVIDUAIS

Onde é permitido criar elementos do diagrama de classe unicamente para quem está operando a aplicação, não afetando assim o projeto colaborativo. Essas funcionalidade não são alvos da política de colaboração.

6.12. FUNCIONALIDADES COLABORATIVAS QUE AFETAM A ÁREA DE TRABALHO LOCAL E PODEM AFETAR O COLETIVO

Essas funcionalidades podem ser executadas sem prejuízo para o projeto colaborativo, mas dependendo da ação posterior podem afetar o ambiente como um todo. Pode-se citar:

- criar uma classe;
- fazer uma associação entre classes;
- excluir uma classe não publicada;
- visualizar a área de trabalho de outro usuário;

- fazer uso da comunicação direta através do *chat*[6]*;*
- salvar uma cópia da área de trabalho localmente.

6.13. FUNCIONALIDADES COLABORATIVAS

São funcionalidades que afetam diretamente o ambiente colaborativo, como por exemplo:

- publicar um elemento;
- voltar um elemento publicado ao estado embrionário;
- convocar uma tomada de decisão para aceitação do elemento recém-publicado;
- inserir novas informações (atributos e operações) em uma classe publicada;
- organizar as classes em sua área de trabalho.

6.14. FUNCIONALIDADES COLABORATIVAS INTERNAS

São funcionalidades automatizadas, internas do ambiente colaborativo, que são executas incondicionalmente, ou seja, não dependem de ação direta dos usuários. São necessários para manter o ambiente organizado e sincronizado. Essas funcionalidades podem ser executadas em períodos de tempo pré-determinados, ou em decorrência de um outro evento, e são:

- salvar o diagrama colaborativo periodicamente;
- apurar término de votação;
- manter todos os diagramas locais atualizados e sincronizados com o a versão do servidor.

[6] Aplicativo ("bate-papo"), onde duas ou mais pessoas utilizam para se comunicarem.

7. IMPLEMENTAÇÃO DA POLÍTICA PROPOSTA (PROTÓTIPO)

Este capítulo descreverá a implementação da proposta da política de colaboração apresentado no capítulo 5 e tem como objetivo validar e testar os requisitos da política de colaboração. A política de colaboração foi implementada em termos de regras e diretrizes parametrizadas no software (aplicativo), possibilitando que cada coordenador de projeto customize a política de colaboração, de acordo com as necessidades de cada projeto, e que melhor se adapte ao contexto de cada grupo de trabalho.

A organização deste capítulo dar-se-á da seguinte forma:

- contexto físico do ambiente colaborativo, destacando o servidor, que é a parte centralizadora do ambiente;
- ilustração de uma instância da arquitetura de distribuição do ambiente colaborativo;
- instalação/configuração do lado servidor e do papel do coordenador geral do projeto, cadastrando os projetos e seus respectivos coordenadores;
- instalação do cliente e o papel do coordenador de projeto, configurando a política de colaboração para o seu projeto;
- instalação do cliente e o papel dos modeladores que atuarão na modelagem dos projetos;
- descrição completa do ciclo da criação dos objetos (classes e associações) e a aplicação das funcionalidades da política proposta;
- acompanhamento do trabalho, pelo coordenador de projeto e pelos modeladores.

7.1. CONTEXTO FÍSICO DO AMBIENTE COLABORATIVO

Como forma de oferecer suporte à política em ambiente colaborativo, uma arquitetura de rede de computadores tem um papel fundamental. Para esse contexto do ambiente colaborativo, a arquitetura baseia-se nas aplicações Cliente/Servidor, que segundo Lower (1995), divide-se em três componentes básicos: um cliente, um servidor e uma rede conectando o cliente ao servidor, conforme ilustra a Figura 25. Nessa arquitetura, tem-se o servidor, onde todos os serviços de acessos remoto ficam centralizados, e os clientes (modeladores) que farão acessos remotos aos recursos disponíveis no servidor. Essa arquitetura possibilita a comunicação entre os modeladores, o controle de acesso aos recursos (serviços) considerados concorrentes, bem como uma administração organizacional do ambiente.

O servidor é utilizado para sustentar o ambiente colaborativo, dando suporte e oferecendo serviços ao cliente. O cliente, por sua vez, utiliza-se de recursos disponíveis no servidor, chamados de recursos compartilhados, que são comuns a todos os clientes. A comunicação via protocolo TCP/IP é utilizada na conexão entre o cliente e o servidor. No lado do servidor ficam disponíveis alguns serviços de segurança e de concorrência aos recursos considerados críticos, proporcionando um suporte de estabilidade ao ambiente colaborativo.

Muitos dos serviços que não afetam o grupo de trabalho, mas apenas a estação local, são oferecidos nas próprias estações de trabalho dos clientes (modeladores), não sobrecarregando assim a comunicação (tráfego de dados). A comunicação baseada no TCP/IP (*internet*), além da própria comunicação, oferece a identificação dos clientes com o servidor através do seu número do IP. Uma vez conhecido o IP do Servidor e esse estando ativo, em qualquer lugar onde se tem disponível ao acesso à *Internet*, a comunicação é estabelecida entre os clientes e o servidor, proporcionado assim o ambiente colaborativo. A característica da comunicação entre os atores desse ambiente colaborativo é síncrona, pois oferece mecanismos eficientes de controle de

concorrência e de atualizações automáticas, permitindo a comunicação em tempo real.

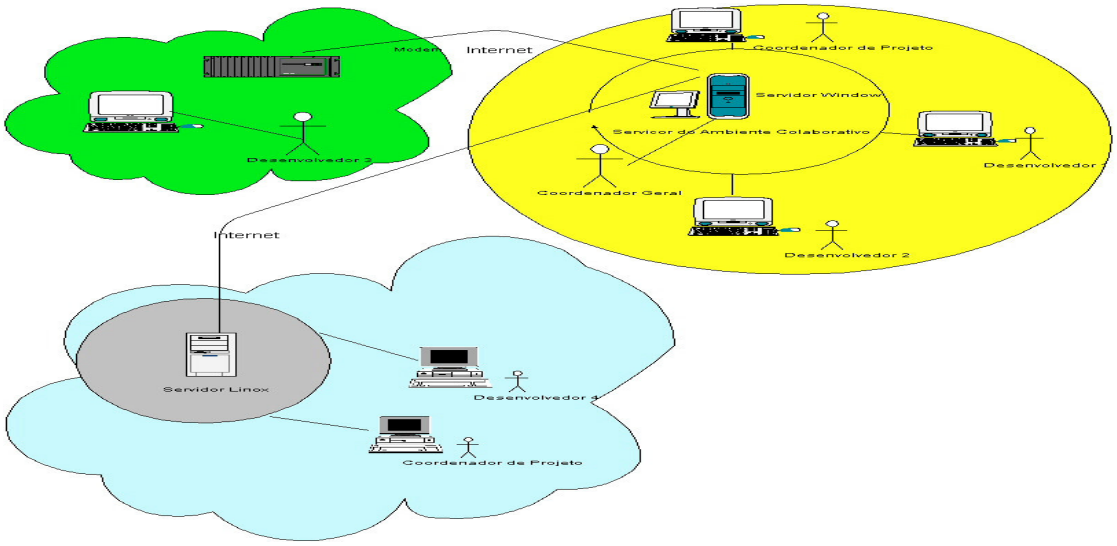


FIGURA 25 - UMA INSTÂNCIA DE ARQUITETURA DE DISTRIBUIÇÃO DO AMBIENTE COLABORATIVO

Na Figura 25, pode-se observar três possíveis contextos de configuração do ambiente colaborativo. Na área de cor amarela, tem-se a instalação do servidor, possibilitando o trabalho do coordenador geral, e alguns clientes (coordenador de projeto, modeladores 1 e 2), conectados em rede local, estando geograficamente no mesmo local e ao mesmo tempo. Na área de cor verde, tem-se um outro cliente (desenvolvedor 3), que faz acesso ao servidor via *modem/internet*, estando em local diferente, porém ao mesmo tempo. Na área de cor azul, há outros clientes (coordenador de projeto e o desenvolvedor 4) conectados em rede local, a um servidor, que por sua vez conecta-se via *internet* ao servidor da área de cor amarela, onde encontram-se os serviços e recursos do ambiente colaborativo da ferramenta. Enquanto na área amarela os clientes ocupam um mesmo local físico, os demais (áreas verde e azul) estão geograficamente dispersos, caracterizando assim um ambiente colaborativo distribuído.

7.2. A INSTALAÇÃO/CONFIGURAÇÃO DO LADO SERVIDOR E O PAPEL DO COORDENADOR GERAL DO PROJETO

Para dar vida ao ambiente colaborativo, alguns procedimentos devem ser realizados, sendo que o primeiro deles é a escolha de um equipamento que esteja com as seguintes configurações:

- estar ligado (conectado) em rede (*internet*);
- com SO Windows/Linux/Unix;
- ter uma configuração de IP com uma numeração fixa;
- possuir uma versão da arquitetura JAVA a partir da *j2sdk-1.4.2;*
- diretório contendo o aplicativo.

Estando satisfeitos os requisitos necessários para a escolha de um servidor, o próximo procedimento a ser realizado é a ativação de dois serviços:

- *RMIREGISTRY*, que basicamente conterá as referências dos serviços oferecidos pelo aplicativo no servidor, possibilitando o acesso remoto dos clientes ao servidor;
- aplicativo (lado servidor) de serviços de acesso remoto, que contém os serviços propriamente dito.

Após o aplicativo do lado servidor da ferramenta ter sido carregado, o funcionamento do primeiro estágio de suporte ao ambiente colaborativo é realizado. Nesse momento, sendo a primeira vez que a ferramenta é instalada no servidor, ocorrerá a verificação se há um coordenador geral cadastrado. Caso a resposta seja negativa, um alerta com a mensagem informando que não há coordenador geral cadastrado é exibido. É então solicitada a confirmação do cadastramento, conforme ilustra a Figura 26.

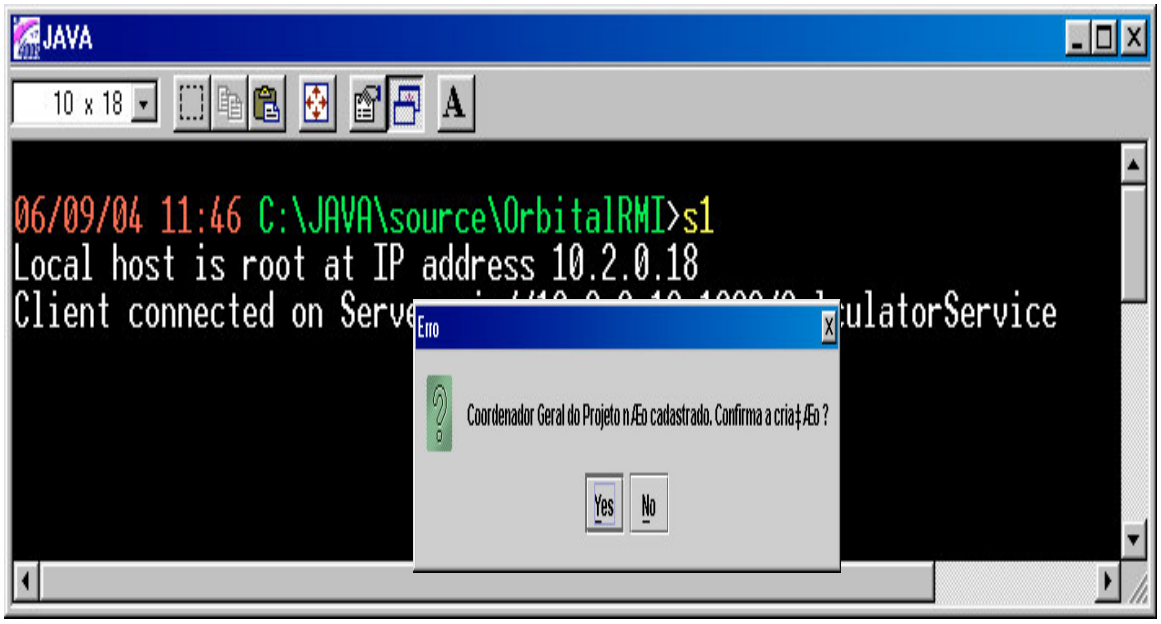


FIGURA 26 - APLICATIVO NO SERVIDOR COM UM ALERTA INFORMANDO QUE NÃO EXISTE O COORDENADOR GERAL CADASTRADO.

Ao escolher-se a opção *YES* (sim), será exibida a interface abaixo, figura 27, solicitando dados para que o cadastramento seja feito. Caso a resposta seja *NO* (não), o aplicativo é encerrado, pois não pode operar sem que haja um coordenador geral.

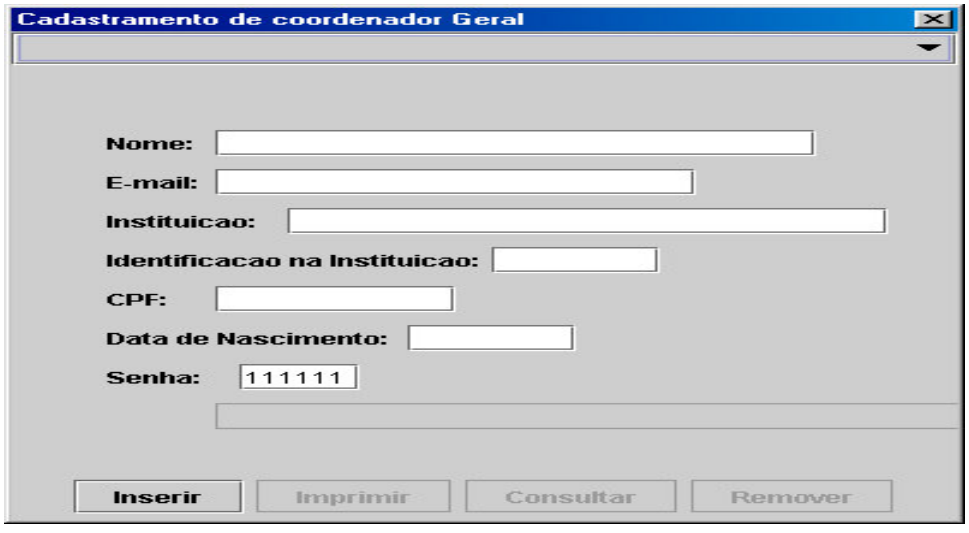


FIGURA 27- INTERFACE DE CADASTRAMENTO DE USUÁRIOS

O cadastramento do coordenador geral constitui a primeira etapa da identificação dos tipos de usuários que foi previsto na política de colaboração. É a partir desse usuário que os demais usuários poderão ser reconhecidos no ambiente colaborativo, possibilitando a aplicação da política proposta.

Uma vez cadastrado o coordenador geral, a versão do cliente poderá ser executa nessa primeira fase. Na entrada da versão cliente, há a necessidade de se informar a localização do servidor com o número do IP, o nome do usuário e a senha de acesso.

FIGURA 28 - INTERFACE DE IDENTIFICAÇÃO DO CLIENTE PARA ACESSO AO APLICATIVO

Essa interface, figura 28, será requerida para qualquer usuário que queira utilizar a ferramenta. É através dessa interface que a ferramenta terá condições de identificar, primeiramente, se o IP do Servidor é válido e se o mesmo está ativo, e, em seguida, se o usuário que está solicitando a entrada na sessão é um usuário válido. Em caso positivo receberá, então, os devidos direitos de acesso.

A próxima etapa para a configuração do ambiente colaborativo é configurar o ambiente com os coordenadores e seus respectivos projetos. A interface, conforme ilustra a Figura 29, permite que somente o coordenador geral faça o acesso. O cadastramento do projeto é feito mediante o acesso à opção *Cadastrar Projeto, Cadastrar Coordenador de Projeto.* A opção

Ambiente colaborativo fica indisponível para o coordenador geral, pois é uma opção restrita somente ao coordenador de projeto.

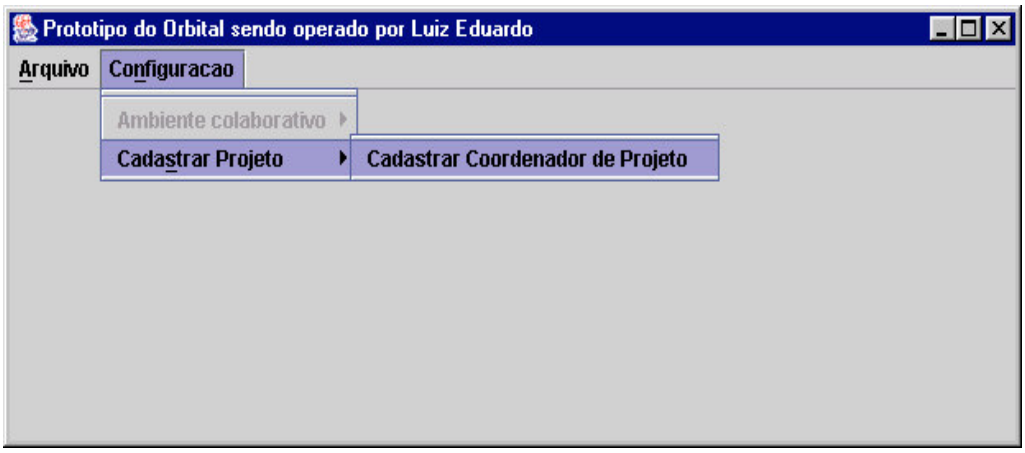


FIGURA 29 - MENU DE CONFIGURAÇÃO DO AMBIENTE COLABORATIVO RESTRITO AO COORDENADOR GERAL

Ao acessar a opção *Cadastrar Projeto,* duas interfaces serão exibidas para completar a configuração, sendo a primeira o cadastramento dos dados dos coordenadores de projetos, conforme ilustra a Figura 29, a interface de cadastramento dos projetos e a vinculação do projeto aos coordenadores cadastrados, como ilustra a Figura 30.

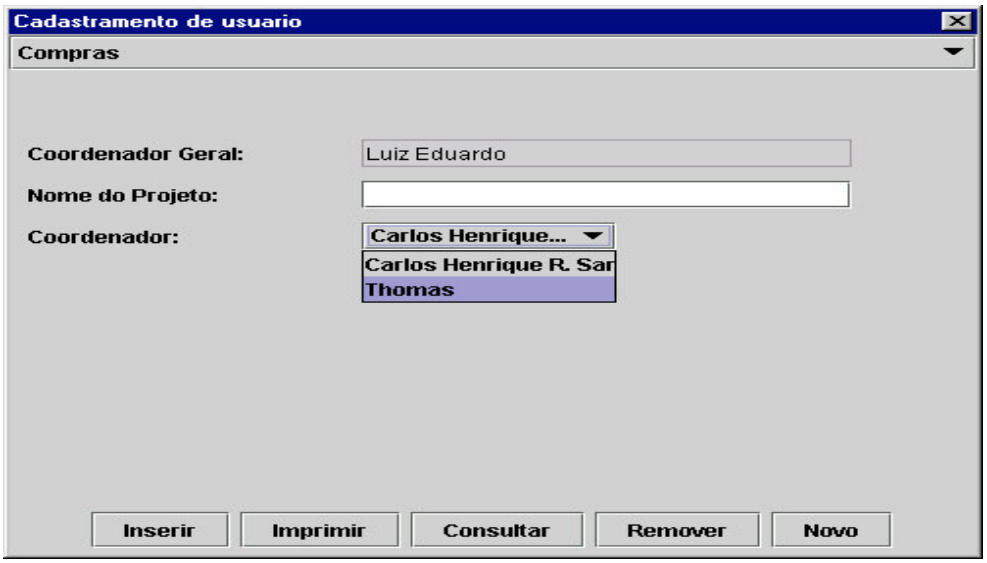


FIGURA 30 - INTERAFACE DE CADASTRAMENTO DE PROJETO E VINCULAÇÃO DE COORDENADORES AO PROJETO

Feitos os cadastramentos dos coordenadores de projetos, o próximo passo é informar o nome de cada projeto e vinculá-lo a um

coordenador, finalizando assim a etapa de configuração do ambiente colaborativo e a atuação do coordenador geral do projeto. Com isso, pode-se iniciar a próxima fase de configuração, que será feita pelos coordenadores de cada projeto.

7.3. O PAPEL DO COORDENADOR DE PROJETO NA CONFIGURAÇÃO DO AMBIENTE COLABORATIVO

A segunda etapa de configuração do ambiente colaborativo é a customização da política de colaboração, por projeto, ou seja, cada coordenador tem à sua disposição, através da interface do aplicativo, possibilidade de escolher determinadas opções (critérios/diretivas) da política de colaboração. Com isso, cada projeto pode ter uma configuração de política de colaboração diferenciada também. A diferenciação está ligada às necessidades dos coordenadores de projeto, possibilitando controlar e organizar o ambiente. Para a viabilização da política de colaboração diferenciada, o coordenador de projeto tem disponível, no menu da ferramenta, as seguintes opções, conforme ilustra a Figura 31.

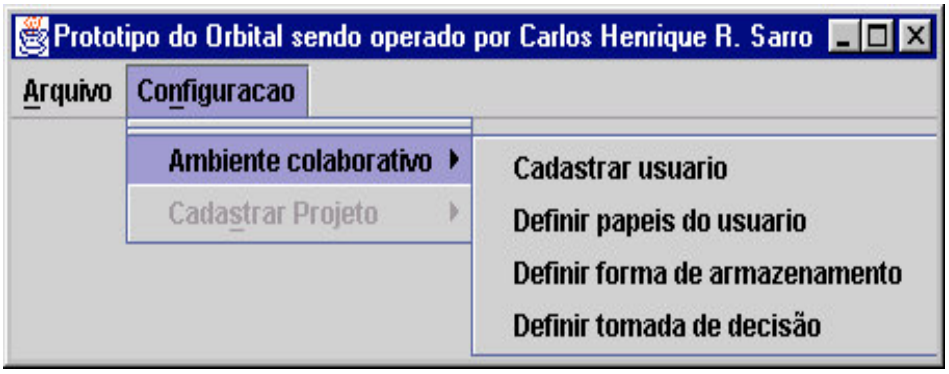


FIGURA 31 - OPÇÕES DE CONFIGURAÇÕES DA POLÍTICA DE COLABORAÇÃO PARA O PROJETO

O acesso ao aplicativo dar-se-á da mesma forma, como foi feito pelo coordenador geral, conforme ilustra a Figura 28. Após o acesso e a identificação de que o usuário é um coordenador de projeto, o menu com as opções *Configuração, Ambiente colaborativo* ficará disponível. Como esse

usuário é um coordenador de projeto, a opção *Cadastrar Projeto* fica indisponível, pois é restrito apenas ao coordenador geral.

A configuração do ambiente colaborativo tem as opções descritas a seguir.

7.4. CADASTRAR USUÁRIOS

Na opção *Cadastrar usuário* o coordenador de projeto tem a possibilidade de formar o grupo de modeladores que atuarão na modelagem do projeto que está sob sua responsabilidade. O cadastramento desses usuários é feito através da interface, conforme ilustra a Figura 27.

7.5. DEFINIR PAPÉIS DO USUÁRIO

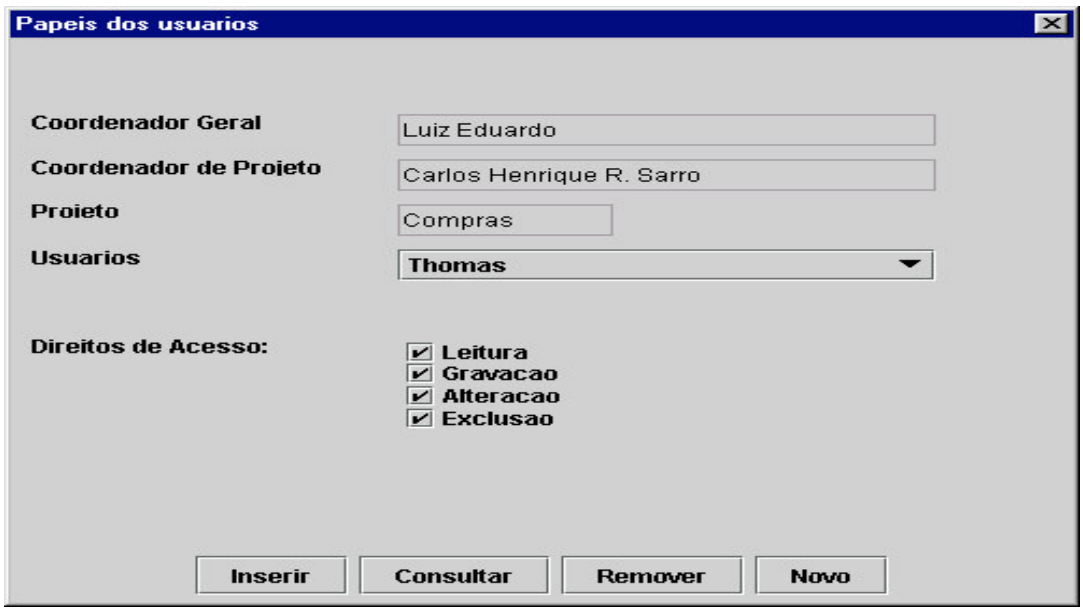


FIGURA 32 - INTERFACE DE ATRIBUIÇÃO DE DIREITO DE ACESSO AOS USUÁRIOS MODELADORES

Com a opção *Definir papéis de usuário*, o coordenador atribui direitos como: *Leitura, Gravação, Alteração e Exclusão*, conforme ilustra a Figura 32. Esses atributos permitirão aos modeladores atuarem sobre os elementos (classes e associações, atributos etc.) durante a fase de

modelagem. Os atributos de direitos de acesso são carregados no início da sessão, quando o usuário faz o *logon,* conforme ilustra a Figura 28. Qualquer modificação feita pelo coordenador de projeto nos direitos de acesso de um determinado desenvolvedor, será refletido imediatamente no ambiente (*on the fly*), mesmo que o usuário esteja em plena atividade na sessão.

7.6. DEFINIR FORMA DE ARMAZENAMENTO

Conforme ilustra a Figura 33, o coordenador de projeto pode definir qual a opção de armazenamento do seu projeto. As opções disponíveis são:

- banco de dados: dá suporte ao armazenamento de estruturas de dados orientado à objeto, *CACHE*;
- vector: um tipo de classe na linguagem JAVA capaz de armazenar objetos, a qual contém todos os dados do projeto.

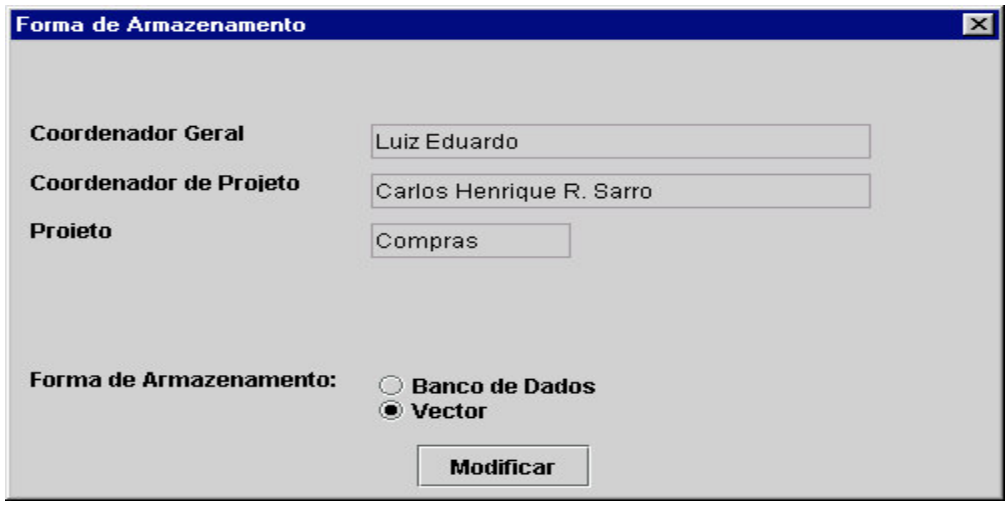


FIGURA 33 - INTERFACE PARA DEFINIR A FORMA DE ARMAZENAMENTO

Em ambas as opções, *Banco de Dados* ou *Vector*, o armazenamento ocorrerá sempre no servidor, garantindo o controle das atualizações, pois são controladas internamente pelo ambiente, ou seja, um auto-salvamento. Com isso, não há preocupação por parte do coordenador do projeto e dos modeladores quanto à periodicidade do salvamento do mesmo,

por quem será realizado, e o local do armazenamento, ficando assim transparente quanto à localização das informações.

7.7. DEFINIR TOMADA DE DECISÃO

O coordenador do projeto tem através da opção '*Definir tomada de decisão',* conforme ilustra a Figura 34, três opções de configuração para adotar no projeto, que são:

- **eleição simples:** quando 50% mais um dos usuários cadastrados no projeto aceitam a mudança de estado do objeto. A eleição é encerrada quando o percentual é atingido ou quando o tempo estipulado para que todos votem é expirado;

- **autoridade do coordenador:** somente o coordenador do projeto decide, aceitando ou não, a mudança de estado do objeto. Pode ser utilizado como saída quando há um empasse muito grande entre os colaboradores, já que o coordenador do projeto tem um papel de moderador durante a sessão;

- **consenso:** todos os usuários presentes à sessão podem votar. É considerada aceita ou não a mudança de estado do objeto quando todos os que votaram têm a mesma opinião, ou seja, todos votaram pelo *sim,* ou todos votaram pelo *não*.

Nessa opção um fator que pode ser um complicador, é o parâmetro de '*prazo limite para votar elemento'*, pois, para os que estão presentes na sessão, se somente alguns votarem, o elemento será aceito sem que haja a apreciação pelos demais modeladores presentes.

Com o acesso a todas as opções de configuração pelo coordenador de projeto, escolhendo qual a configuração adequada para o projeto a ser desenvolvido pelo modeladores, encerra-se a fase de configuração do ambiente colaborativo. Com isso, permite-se que todos os

modeladores tenham condições de acessar a ferramenta e entrar na sessão para realizar a modelagem.

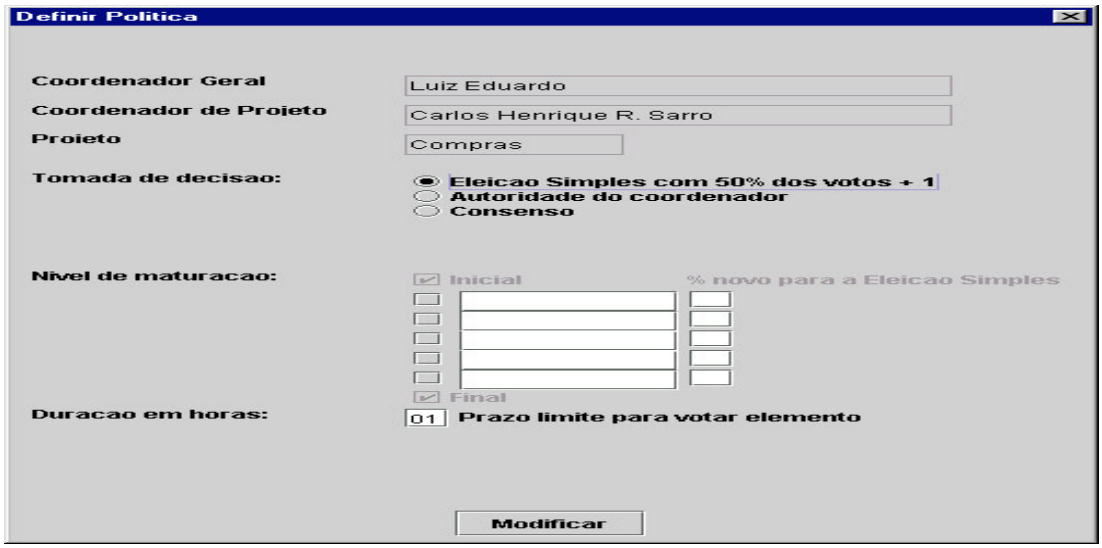


FIGURA 34 - INTERFACE PARA DEFINIÇÃO DA POLÍTICA QUANTO À TOMADA DE DECISÃO E NÍVEIS DE MATURAÇÃO

Sempre que o coordenador do projeto julgar necessário que alguma configuração seja alterada, ajustando à necessidade do projeto, o acesso às opções de configuração ficarão disponíveis, independente do local onde o acesso é realizado. As alterações realizadas no ambiente colaborativo, pelo coordenador do projeto, terão efeito imediato nas ações que estão ocorrendo.

7.8. O PAPEL DO DESENVOLVEDOR NA EDIÇÃO DE UM PROJETO COLABORATIVO

Na última fase da configuração do ambiente colaborativo é feita a instalação de uma cópia do aplicativo (cliente) em cada máquina dos modeladores, que configuradas com o número do IP do servidor, para que seja possível a conexão com o ambiente colaborativo.

7.9. DEFINIR TOMADA DE DECISÃO

Sempre que o aplicativo for carregado, o número do IP do servidor deve ser informado, conforme ilustra a Figura 28. Nessa ocasião é possível também que um outro endereço de IP seja informado, caso haja uma mudança ou queira-se fazer a conexão com um outro servidor.

A conexão com o servidor e a identificação do usuário no *logon* de acesso da ferramenta são partes dentro do contexto do ambiente colaborativo. Nessa operação são validados se o servidor está disponível e se o usuário que está solicitando a entrada é válido para o ambiente.

Em se estabelecendo a conexão com o servidor, a interface principal da ferramenta é apresentada, conforme ilustra a Figura 35. Nessa interface é possível ter acesso à opção de *Arquivo*, *Abrir Diagrama Colaborativo, Projeto*.

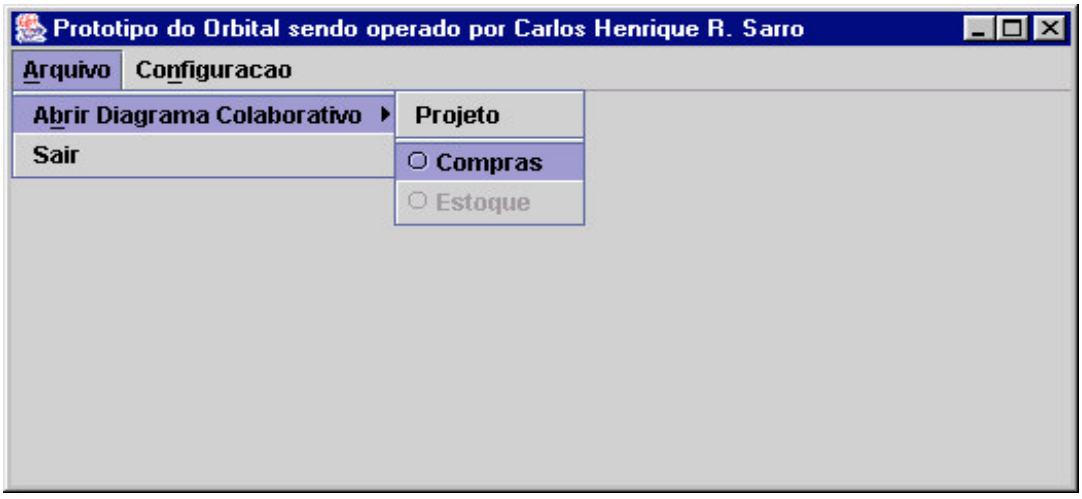


FIGURA 35 - MENU PRINCIPAL DA FERRAMENTA. OPÇÕES DE ESCOLHA PARA A ABERTURA DE UM DIAGRAMA COLABORATIVO

Acessando a opção de *Projeto*, todos os projetos cadastrados no ambiente são apresentados, sendo, nesse exemplo, os projetos de 'Compras' e 'Estoque'. Porém, fica acessível somente o projeto no qual o

usuário tem o direito de acesso, que neste caso é o 'projeto de compra'. As opções de configurações ficarão disponíveis somente para os coordenadores, evitando assim qualquer interferência não desejada no ambiente, caso o usuário seja o coordenador geral ou um desenvolvedor.

7.10. EDITANDO UM PROJETO COLABORATIVO

Escolhendo o projeto em que se tem direito de acesso, uma interface é apresentada com vários ícones de suporte à edição do diagrama de classes, conforme ilustra a Figura 36. Todos os usuários que estão presentes no ambiente colaborativo são identificados e seus nomes e fotos (quando existirem) são exibidos nas abas localizadas na parte inferior da interface. Esse efeito também ocorre em todas as estações de trabalho, ou seja, a entrada na sessão por um determinado usuário provocará a atualização, a inclusão de mais uma aba na interface. A saída de um deles provoca a exclusão da aba de todas as estações de trabalho.

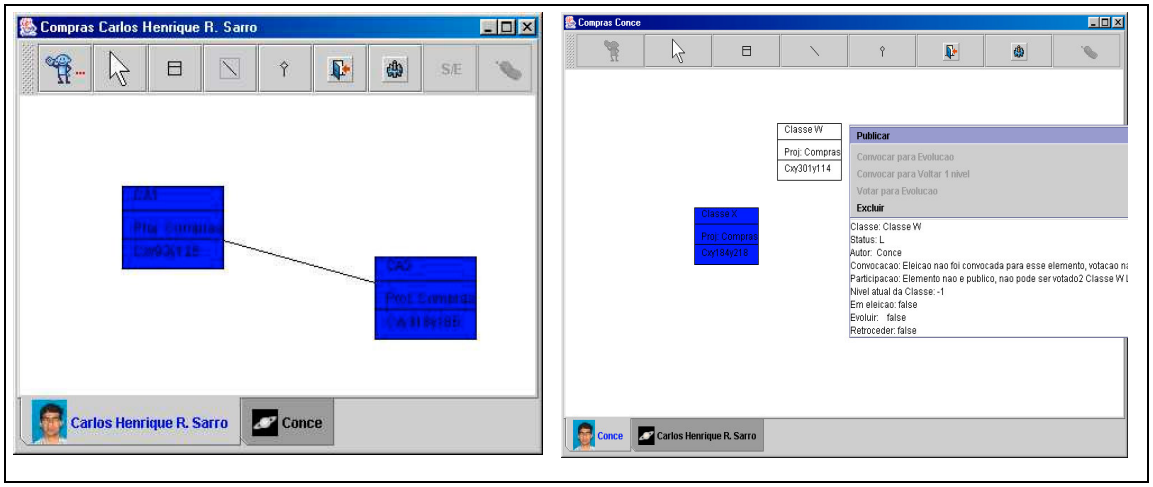


FIGURA 36 - INTERFACE DE EDIÇÃO COLABORATIVA DO DIAGRAMA COM ACESSO À ÁREA DE TRABALHO DOS DEMAIS MODELADORES E FERRAMENTA DE MODELAGEM DENTRO DAS ESPECIFICAÇÕES DA UML.

Sempre que o desenvolvedor desejar saber o que outros usuários estão realizando, basta clicar na aba do desenvolvedor escolhido para que seja possível a visualização de sua área de trabalho, implementando dessa forma o conceito de (*WYSIWIS*). Porém o acesso à área de trabalho de

outro desenvolvedor fica restrito somente para a visualização, possibilitando o conhecimento entre eles.

Ao criar um elemento do diagrama de classe (conforme ilustra a Figura 40 a 42 da sessão do apêndice), esse permanece somente na área de trabalho do desenvolvedor que a criou, permitindo a realização de qualquer ação nesse elemento. Nesse estágio, o elemento é classificado como estado embrionário, onde o desenvolvedor poderá fazer qualquer alteração, movimentação (arrastar com o *mouse*) ou mesmo excluir esses elementos, pois não afetará o diagrama, que é comum à sessão de trabalho colaborativa. Todavia, esses elementos não publicados poderão ser visualizados por um outro desenvolvedor que esteja ativo na colaboração sem, porém, acesso habilitado. Uma facilidade dessa abordagem é a liberdade de ações que o desenvolvedor tem para fazer simulações, sem afetar o trabalho do grupo.

7.11. PUBLICAR UM ELEMENTO

Para que o elemento criado, que está no estado embrionário, possa fazer parte do diagrama colaborativo e ser submetido à aprovação, conforme opção adotada na política de tomada de decisão, é preciso que seja publicado. A publicação é uma ação disponível somente aos elementos que estão no estado embrionário. A ação de *publicar* um elemento pode ser feita acionando-se diretamente o ícone da barra de ferramenta (sexto ícone), ou acessando-se o menu de opções pressionando o botão direito do *mouse,* posicionado em cima do elemento. Após a publicação do elemento, este passa a estar disponível na área de trabalho de todos os modeladores.

Caso ocorra uma publicação indevida do elemento, o autor poderá voltar o elemento publicado ao estado embrionário, acessando o menu de opções pressionando o botão direito do *mouse.*

7.12. RECONHECER UM ELEMENTO PUBLICADO

Para que os novos elementos que vão sendo publicados possam ser reconhecidos pelos demais modeladores, um mecanismo de aviso (*gif* animado) é habilitado na barra de ferramenta, avisando que há novos elementos publicados e disponíveis para serem incorporados na área de trabalho. Esse mecanismo garantirá que não haverá sobreposição com elementos que estão chegando à sua área de trabalho. Esse mecanismo possibilita também que as alterações nos elementos alocados sejam finalizados antes da atualização (*refresh)* na tela. Para facilitar a distinção dos elementos públicos com os demais que ainda estão no estado embrionário, os elementos passam a ter uma cor em tom azul.

7.13. CONVOCAR UMA TOMADA DE DECISÃO PARA MUDANÇA DE FASE DE MATURAÇÃO DE UM ELEMENTO

Uma vez que um elemento tenha sido publicado na área comum de trabalho, precisa ser aceito pelos demais modeladores do grupo de trabalho como sendo importante e que contribuirá para o modelo final. Para isso, é necessário que o autor da criação do elemento recém-publicado convoque uma tomada de decisão. A política da tomada de decisão a ser aplicada é definida pelo coordenador do projeto, podendo ser: a *Eleição Simples, Consenso ou Autoridade do Coordenador.* A ação de convocação é feita pressionando-se o botão direito do *mouse* e escolhendo-se a opção *convocar para evoluir*, conforme ilustra a Figura 43 (A). De igual modo uma convocação precisa ser feita quando um elemento que é público, e que já sofreu uma evolução, precisa ser retirado do diagrama ou retroceder um nível. Semelhantemente ao que é feito para os elementos publicados, um dispositivo de aviso é ativado na barra de ferramenta do aplicativo de cada modelador, informando que há uma eleição a ser realizada. Caso a tomada de decisão seja a *autoridade do coordenador,* somente o coordenador do projeto recebe o aviso.

Quando o elemento é convocado para uma tomada de decisão, a ação de alocação para incluir, alterar ou excluir atributos e operações é desativada. Tão logo se encerra o processo de apuração, a alocação passa a ficar disponível novamente.

7.14. VOTAR

Realizada a ação de convocação, a opção de votar para evoluir ou retroceder passa a ficar habilitada como item de menu (opção pode ser acessadas pressionando o botão direito do *mouse*). Ao acionar-se a opção de votação, três opções de voto são exibidas na interface do aplicativo, sendo: *Sim, Não, Abstenção.* Essas três opções são disponibilizadas quando a política de *tomada de decisão* estiver configurada para *Eleição Simples* e *Consenso.* Quando a política de *tomada de decisão* for *Autoridade do Coordenador,* somente as opções *Sim e Não* ficam disponíveis, pois nesse caso a aceitação de evoluir ou retroceder fica só na responsabilidade do coordenador do projeto. Para as tomadas de decisão *Eleição Simples* e *Consenso,* todos os integrantes do grupo de trabalho, inclusive o coordenador do projeto, têm direito de voto.

Uma vez que o modelador vote, a opção de votação torna-se-lhe indisponível, não permitindo que se vote novamente. A desabilitação da opção de votação também é acionada quando há o encerramento da eleição ou quando o tempo limite de votação, parametrizado pelo coordenador do projeto, excede.

7.15. APURAR

O processo de apuração é ativado no mesmo momento em que a ação de convocação é acionada. Durante a votação o processo de apuração realiza as seguintes operações:

- garantir que um modelador vote somente uma única vez, quando a tomada de decisão for *Eleição Simples* e *Consenso;*

- encerrar a eleição imediatamente quando o coordenador de projeto votar, quando a tomada de decisão for *Autoridade do Coordenador;*

- emitir resultados parciais do processo da apuração quando solicitado pelo coordenador do projeto;

- encerrar a eleição e apurar o resultado quando o tempo de votação excedeu ao limite previamente estabelecido pelo coordenador de projeto;

- encerrar a apuração quando a tomada de decisão for *Eleição Simples* e atingiu o percentual mínimo de aprovação ou rejeição (50% mais um dos votantes);

- quando o resultado da apuração for positivo, a eleição é dada por encerrada e o elemento, ou é elevado a um estado acima, ou retrocede para um estado anterior, dependendo do tipo de convocação que foi solicitado.

O monitoramento constante garante que o processo de eleição seja conclusivo e eficaz. Com isso, o elemento que estava desabilitado para alterações passa a ser novamente liberado aos modeladores.

7.16. ALOCAR ELEMENTO

Sempre que um modelador fizer um acesso a um elemento, pressionando o botão direto do *mouse,* nas opções *atributos* e *operações,* o mecanismo de alocação é ativado (XU, 2005). A alocação é feita através do algoritmo de exclusão mútua. Com isso, é possível garantir que somente um modelador por vez faça atualizações sem que haja sobreposição, evitando perda de trabalho.

Ao acessar-se um elemento que está alocado por um outro modelador um aviso é emitido. Nesse aviso é informado o nome do modelador

que fez a alocação, permitindo que o interessado visualize a área de trabalho e negocie através do *chat* a liberação do elemento alocado.

Caso um modelador faça o acesso a um determinado elemento e o tempo de alocação exceda em uma hora, sem que haja nenhuma modificação, o elemento é liberado automaticamente sem prévio aviso, para outros modeladores fazerem o acesso.

O acesso a um elemento alocado feito via botão direito do *mouse*, não é impedido, pois é garantido a obtenção de informações pertinentes ao estado atual do elemento no ambiente colaborativo.

7.17. MECANISMO DE COMUNICAÇÃO

A colaboração fica mais enriquecida com o mecanismo de comunicação presente na barra de ferramentas, conforme ilustra a Figura 46, que é o *Chat* (ícone do telefone), onde os modeladores têm condições de resolver impasses, oferecer ajuda e provocar a ação de questões pendentes na edição do diagrama, conforme ilustra a Figura 37.

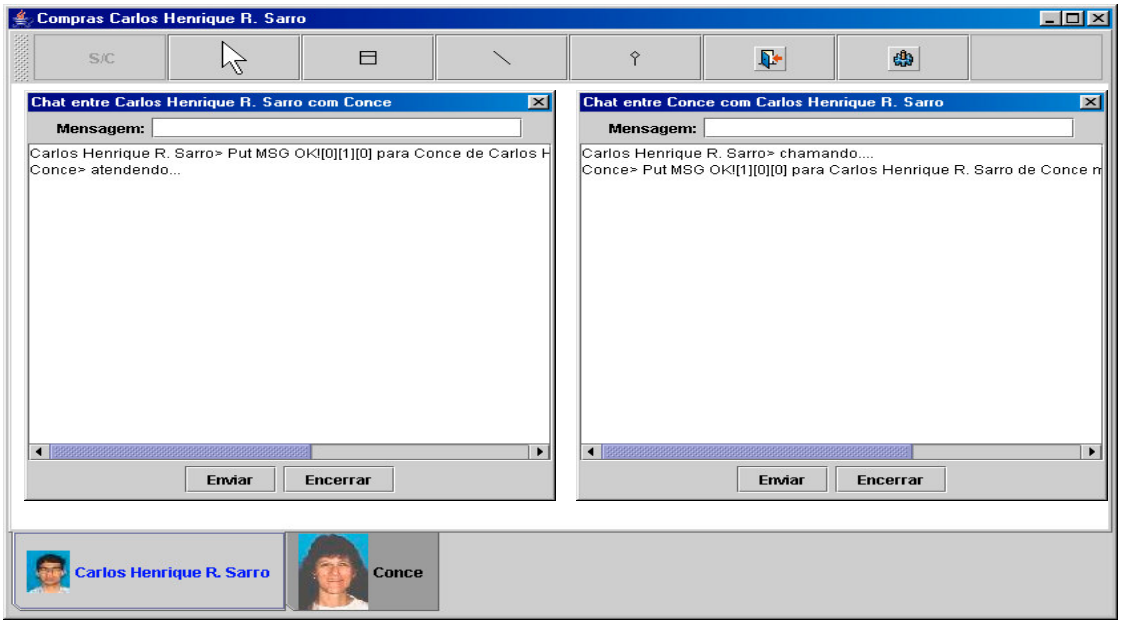


FIGURA 37 - INTERFACE DO CHAT, ONDE DEMONSTRA A COMUNICAÇÃO ENTRE DOIS MODELADORES DE UM MESMO PROJETO

7.18. ACESSAR INFORMAÇÕES DE UM ELEMENTO

Para acessar as informações implícitas de um elemento, basta o desenvolvedor pressionar o botão direito do *mouse.* No acesso é possível que se saiba (vide Figura 36 (B)):

- o autor da criação do elemento;
- em que fase de maturação está no exato momento;
- se houve convocação para a tomada de decisão;
- qual o resultado parcial da apuração quando o elemento for acessado pelo coordenador de projeto;
- se a convocação é para evolução do elemento ou retroceder um nível;
- saber se o modelador que indagou a consulta votou ou não quando o elemento esteve em votação.

7.19. TESTE DO PROTÓTIPO IMPLEMENTADO

Durante a fase de validação da proposta, foram realizados testes com o protótipo implementado, onde os desenvolvedores interagiam no ambiente colaborativo, na realização da modelagem do diagrama de classes.

As seguintes situações relatadas abaixo mostram o desempenho do ambiente colaborativo, quanto ao tempo de resposta.

1. Instalação do ambiente colaborativo do lado do servidor e dos desenvolvedores no mesmo ambiente (sala) físico, ou seja, os desenvolvedores e o servidor não estavam geograficamente dispersos e sim compartilhavam do mesmo ambiente físico e lógico da rede;

2. Instalação do servidor colaborativo em ambiente separado aos dos desenvolvedores, porém, dentro do mesmo prédio. Envolvia esta situação apenas roteadores e *hubs*;

3. Instalação do servidor em ambiente geograficamente diferente dos clientes. A comunicação entre os dois ambientes é feita via rádio a 10 mb;

4. Acesso remoto da residência de um dos desenvolvedores via rede *dial up* (discagem via telefone).

Nas situações 1 e 2, o desempenho do ambiente colaborativo mostrou-se com boa performance, ou seja, as respostas formam imediatas.

Na situação 3, foi visível o retardo das respostas do ambiente colaborativos às ações dos desenvolvedores, porém, não foi prejudicial ao trabalho de modelagem.

Na situação 4, o tempo de resposta foi extremamente alta e inviabilizando assim a modelagem colaborativa.

Conclui-se de fato, que o desempenho da comunicação é um fator crítico de sucesso em ambientes colaborativos. Quanto menor o tempo de respostas, maior será o grau de satisfação dos usuários desenvolvedores.

Os testes de verificação dos requisitos funcionais e não funcionais se basearam nos seguintes itens do *check-list:*

- ✓ Cadastramento de usuários;
  - ❑ Coordenador geral;
  - ❑ Coordenador de Projeto;
  - ❑ Desenvolvedores.
- ✓ Cadastramento de Projeto;
  - ❑ Atribuição do coordenador ;

- Formação da equipe de desenvolvimento;
- Atributos de controle de acesso;
- Política customizada;
  - Forma de armazenamento;
  - Níveis de maturação dos elementos;
  - Definição da tomada de decisão;

✓ Edição do diagrama;

- Área local;
- Área comum;
- Auto *refresh;*
- *Publicação de um elemento;*
- Alocação de um elemento;
- Acionar a tomada de decisão;
- Votação;
- Apuração da votação;
- Visualização da área de trabalho de outro desenvolvedor.

✓ Comunicação

- Acesso à área de outro desenvolvedor;
- *Chat* entre dois desenvolveres.

Outro teste realizado foi em relação a segurança das informações no ambiente colaborativo. Simulou-se a queda do servidor quando vários desenvolvedores estavam ativos no ambiente, fazendo com que

os aplicativos clientes perdessem a comunicação com o servidor. Quando o servidor foi restabelecido, ou seja, tornou-se ativo, a última cópia do diagrama pode ser restabelecida automaticamente, restaurando assim todas as áreas locais bem como a área comum de trabalho.

8. CONCLUSÃO

O estudo da área de pesquisa *CSCW*, tem tido um papel importante no desenvolvimento das atividades humanas, pois observa-se que com o avanço e crescimento da tecnologia das aplicações *groupware,* muitos obstáculos e barreiras, dentre os quais o espaço e o tempo para a realização do trabalho em grupo, estão diminuindo. Com essa tecnologia os colaboradores são estimulados a desenvolver suas atividades aliada aos recursos computacionais disponíveis para comunicação. Preocupações tais como onde estão as informações e quais os melhores recursos a serem disponibilizados no auxílio da realização de suas atividades, são características já integradas a essa tecnologia. Com isso, a realização de trabalho em grupo com suporte da computação, passa a ser mais freqüente, estimulando os pesquisadores da área de *CSCW* a intensificarem pesquisas que resultem em melhores condições e no oferecimento de novos artefatos que auxiliem os colaboradores em ambiente de trabalho colaborativo.

Nos ambientes colaborativos, os integrantes que atuam na realização do trabalho em grupo, muitas vezes estão geograficamente dispersos. As ações a serem executadas devem ser pertinentes ao objetivo do trabalho, como por exemplo a modelagem de sistemas. Preocupação de organização, controles, localização das informações, alocação e liberação dos recursos, não devem sobrecarregá-los. Para isso, passa a ser importante, na área de CSCW, a discussão e a pesquisa sobre política de colaboração,

Na definição da política de colaboração para um ambiente de co-autoria de diagrama de classes, principal assunto tratado neste trabalho, concluiu-se que devam ser levados em consideração as seguintes questões:

- infra-estrutura organizacional do ambiente;
- aspectos técnicos relevantes de estabilidade e segurança do ambiente e das informações;

- definição dos papéis dos usuários;
- direitos de acesso aos elementos;
- administração de projetos: permitindo que vários projetos sejam realizados no mesmo ambiente, resguardado pelo direito de acesso;
- conhecimento e troca de experiência entre os modeladores: proporcionando condições mínimas de comunicação entre os integrantes do grupo de trabalho;
- flexibilização na formação de regras quanto à política de tomada de decisão, oferecendo três formas diferentes (eleição simples, consenso e autoridade do coordenado), sendo que as duas primeiras formas possibilitam até 5 níveis de amadurecimento das propostas feitas pelos modeladores;
- opção quanto à forma de armazenamento: deixando flexível a opção de escolha entre a forma convencional de armazenamento, e a utilização de banco de dados;
- aspectos tecnológicos: organização do ambiente (cliente/servidor), multi-plataforma (JAVA), comunicação síncrona.

Com base nessas questões, uma política de colaboração para um ambiente de co-autoria de diagrama de classes pode ser implementada, dando suporte a apectos importantes inerentes aos ambientes colaborativos. Caso contrário, o dinamismo das interações que ocorrem nesse ambiente, tenderia ao caos, à desorganização, à sobrecarga dos usuários e sujeito a surpresas, que resultariam em dificuldades para o trabalho em grupo.

Visto que uma política de colaboração é importante para o governo de um ambiente colaborativo, ter a possibilidade de personalizá-la (customização), de acordo com a conveniência e necessidades dos coordenadores de projeto, constitui-se um requisito importante e necessário.

Uma preocupação observada neste trabalho foi oferecer ao grupo colaborativo, recursos de acessos aos elementos que compõem o objetivo do trabalho (modelagem de sistema através do diagrama de classes), possibilitando facilidades de interação e obtenção de informações. Com essas facilidades, pôde-se notar no ambiente virtual de trabalho, três instâncias de "espaço" . A primeira instância é a área de trabalho local, possibilitando total liberdade de manipulação, sem preocupações com erros e acertos, que possam interferir no trabalho do grupo. A segunda instância, é a possibilidade de visualizar o que os outros modeladores estão fazendo em suas respectivas áreas de trabalho, permitindo através do dispositivo de comunicação *CHAT*, interferir indiretamente no trabalho de outros colaboradores, bem como se beneficiar com a troca de conhecimentos. Nessa segunda instância, destaca-se a percepção dos colaboradores em saber o que está ocorrendo ao seu "redor", tanto no conhecimento de "quem" e do "que", presentes no ambiente. A última instância é a área de trabalho coletiva, onde toda e qualquer ação individual afeta o coletivo, exigindo uma atuação profunda da política de colaboração.

As ferramentas CASE têm representado um ganho significativo para a área de computação, oferecendo suporte às diversas fases do desenvolvimento de sistemas, proporcionando padronizações e métodos de trabalhos. Com a automação dos processos, muitas das atividades passaram a ser executadas por uma única pessoa. Com isso, a sobrecarga de trabalho e a falta de participação de outros desenvolvedores somaram como um fator negativo e têm sido a causa de muitas empresas não adotarem ou mesmo abandonarem as ferramentas CASE no desenvolvimento de software. Com esse trabalho, pôde-se notar que, se as características e conceitos aplicados nas ferramentas colaborativas fossem abordadas também para as ferramentas CASE, proporcionariam mais participação da equipe no projeto, alívio de carga de trabalho, aumentando o nível de comunicação, independente da localização de cada um, e, um aumento da produtividade da equipe, bem como todos os benefícios que são pertinentes às aplicações *Groupware* (independência de localização e armazenamento, controles de concorrências, etc).

Em trabalhos futuros a ferramenta incluirá as seguintes atividades:

1. Inclusão dos requisitos colaborativos funcionais e não funcionais validados na proposta, transformando assim a ferramenta ORBITAL versão 1.0 mono-usuária, na versão 2.0 colaborativa;

2. Outros mecanismos pertinentes ao ambiente colaborativo, tais como vídeo conferência e serviço de *email* ;

3. Incorporação das funcionalidades do servidor no aplicativo dos desenvolvedores (clientes), juntamente com mecanismos de monitoramento das atividades do servidor. No caso de falha do servidor, um dos aplicativos dos desenvolvedores (cliente), assume o papel provisório do servidor, contribuindo assim com a estabilidade do ambiente;

4. Viabilizar o experimento de outros tipos de comunicação como *Socket,* que possibilite uma análise comparativa de performance de comunicação com o implementado no protótipo via RMI.

Referências

ARAÚJO, R. **CSCW, Groupware & Internet**. Rio de Janeiro: UFRJ, 1995. Disponível em: <http://www.cos.ufrj.br/~renata/**cscw**/sumario.htm>. Acesso em: 15 jan. 2004.

BAFOUTSOU, G.; MENTZAS, G. Review and functional classification of collaborative systems. **International Journal of Information Management**, v.22, n.4, p. 281-305, Aug/2002.

BAILEY, J.; SWIGGER, K.; VANECEK, Michaell: Computer-supported collaborative work and its application to software engineering. In: **Proceedings of the 1995 ACM SIGCPR Conference on Supporting Teams, Groups, and Learning Inside and Outside the IS Function Reinventing IS.** Disponível em: <www.portal.acm.org>. Acesso em: 11 jan. 2004.

BARROS, Lígia. **Suporte a Ambientes Distribuídos de Aprendizagem Cooperativa**. 1994. Tese (Doutorado) – Universidade Federal do Rio de Janeiro (UFRJ), Rio de Janeiro, 1994.

BOOCH, G.; JACOBSON, I.; RUMBAUGH, J. **UML distilled**. Massachusetts: Addison Wesley Longman, 1997.

BOOCH, G.; JACOBSON, I.; RUMBAUGH, J. **UML, guia do usuário.** Rio de Janeiro: Campus, 2000,

BHUSHAN, B.; PATEL, A. Requirements and concept of cooperative system management. **International Journal of Network Management,** v.8, n.3, p. 139 -158, may/jun, 1998.

BORGES, M.R.S.; CAVALCANTI, M.C.R.; CAMPOS, M.L.M. Suporte por computador ao trabalho cooperativo. **XV Congresso da Sociedade Brasileira de Computação. Jornada de Atualização em Informática**, 1995, Canela – RS. Disponível em: <http://penta.ufrgs.br/pgie/sbie99/acac.html>. Acesso em: 15 jan. 2004.

BULLOK, A.; BENFORD, S. An access control framework for multi-user collaborative environments. (1999). **Proceedings of the international ACM SIGGROUP conference on Supporting group**. Disponível em: <http.www.portal.acm.org>. Acesso em: 21 jan. 2004.

CAMOLESI, L; MARTINS, L. E. G.; Specifying powerful rules to govern collaborative environments. **Proceedings of the Ninth International Conference on Computer Supported Cooperative Work in Design, 2005**, IEEE CNF, Disponível em: < http://ieeexplore.ieee.org>. Acesso em: 27 out. 2005, p. 810 - 815 Vol. 2.

Colaboração ou cooperação. Disponível em: <http://www.ricesu.com.br/colabora/n2/artigos/n_2/id03q2.htm>. Acesso em: 11 mar. 2004.

COMO assumir e vencer batalhas. **Informativo Técnico**, n.71, mar/2000. Disponível em: <http://www.revista.unicamp.br/infotec/informacao/inf71.htm>. Acesso em: 20 jan. 2004.

DAMM, C.H.; HANSEN, K.M.; THOMSEN, M.; TYRSTED, M. Creative object-oriented modelling: support for intuition, flexibility, and collaboration in CASE tools. (2000) In: BERTINO, E.(ed.) **ECOOP 2000**. Disponível em: <http://www.ideogramic.com/download/resources/ecoop2000.pdf>. Acesso em: 10 mar. 2004, p.27-43.

DEWAN, P.; SHEN, H. Flexible meta access-control for collaborative applications. **Proceedings of the 1998 ACM conference on Computer supported cooperative work.** Disponível em: <http.www.portal.acm.org>. Acesso em: 21 jan. 2004.

DU LI, R. R. Runtime dynamics in collaborative systems. ACM **Siggroup Conference On Supporting Group Work**, Phoenix, 1999. Disponível em: <http:www.portal.acm.org>. Acesso em: 15 jan. 2004.

DAMM, C.H.; HANSEN, K. M.; THOMSEN, M.; TYRSTED, M. Tool support for cooperative object-oriented design: gesture based modeling on an electronic

whiteboard. **Proceedings Of The SIGCHI Conference On Human Factors In Computing Systems** (2000). Disponível em: <http.www.portal.acm.org>. Acesso em: 11 jan. 2004.

EDWARDS, W.K. Policies and roles in collaborative applications. **Proceedings Of The 1996 ACM Conference On Computer Supported Cooperative Work**. (1996). Disponível em: <http.www.portal.acm.org>. Acesso em: 12 fev. 2004.

ERIKSSON, H.; PENKER, M. **UML toolKit**. New York: Wiley Computer Publishing, 1998, p.1-9.

ELLIS, C.A.; GIBBIS, S.J.; REIN, G.L. Groupware: some issues and experiences. **Communications of the ACM,** v.34, n.1, p.38-58, jan/1991.

FARIAS, C.R.G. **Architectural design of groupware system**: a component based approach. (2002). [CTTIT, Thesis series 01-38], Disponível em: <http://doc.utwente.nl/fid/1364>. Acesso em: 10 set. 2003, p.1-44.

GREENBERG, S.; MARWOOD, D. Real Time as a distributed system: concurrency control and its effect on the interface. **Proceedings of the 1994 ACM Conference**. Disponível em: <http.www.portal.acm.org>. Acesso em: 21 jan. 2004, p. 207-217.

HAAKE,J. M.; HAAKE, A.; BOURIMI, M.; LANDGRAF, B., End-User Controller Group Formation and Access Rigths Management in a Shared Workspaces System, **Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work**. Disponível em: <http.www.portal.acm.org>. Acesso em: 21 jan. 2004, p. 554-563.

HARMON, P.; WATSON, M. **Understanding UML**: the developer's guide, San Francisco: Morgan Kaufmann Publisher, 1998, p. 47-57.

HOFTE, T. H. Working apart together: fundations for component groupware. **Telematica Institute Fundamental Research Serie**, n.1(TI/FRS/001), 1998. Disponível em: <https://doc.telin.nl/dscgi/ds.py/View/Collection-1702>. Acesso em 11 jan. 2004.

How to draw UML diagrams. Disponível em: <http://www.smartdraw.com/resources/centers/uml/uml2.htm#whatclass>. Acesso em: 03 jan. 2004. [Techinical Edition free download].

HUPFER, S; CHENG L.; ROSS, S; PATTERSON, J., Introducing Collaborative into Application Development Environment, **Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work**. Disponível em: <http.www.portal.acm.org>. Acesso em: 21 jan. 2004, p. 21-24.

IGNAT, C.; NORRIE, M., Grouping in Collaborative Graphical Editors, **Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work**. Disponível em: <http.www.portal.acm.org>. Acesso em: 27 jan. 2004, p. 447-456.

KLING, R. Fair information practices with computer supported cooperative work. **Computer-Mediated Communication Magazine**, v.1, n.2, p.5, 1994. Edição eletrônica disponível em: <http://www.december.com/cmc/mag/1994/jun/cscw.html>. Acesso em: 10 mar. 2004.

LENDING, D.; CHERVANY, N.L. CASE tools: understanding the reasons for non-use. **ACM SIGCPR Computer Personnel**. (1998) Disponível em: <http://web.njit.edu/~jerry/sad/Articles/Lending-ACM-SIGCPR-1998.pdf>. Acesso em: 21 jan. 2004.

LENDING, D.; CHERVANY, N.L. The use of CASE tools. **ACM SIGCPR Computer Personnel**. (1998). Disponível em: <http:www.portal.acm.org>. Acesso em: 21 jan. 2004.

LEE, C.R.; TEPFENHART, W. M. **UML e C++**: guia prático de desenvolvimento orientado a objeto. São Paulo: Makron Books, 2001, p. 505-533.

LETTI, G.; LIMA, P.R.T, ALBINO, S.F.; PANIZ, V.L.F. **Ambientes de trabalho e aprendizagem cooperativos.** Disponível em: <http://wwwedit.inf.ufsc.br:1998/Lages/alunos/cscl_cscw.html>. Acesso em: 10 jan. 2004.

LOWE, D. **Client/Server computing for dummies.** New York: Hungry Minds, 1999.

MACEDO, A.A.; NETO, R.F.B.; PIMENTEL, M.G.C. Autoria colaborativa na Web: experiências e reflexões sobre a coWeb. **Revista Brasileira de Informática na Educação (RBIE)**, n.9, set/2001. Disponível em: <http://www.icmc.usp.br/~ale/farra/Rbie2001.pdf>. Acesso em: 20 jan. 2004.

MARTIN, J.; McCLONE, C. **Técnicas estruturadas e CASE**. São Paulo: Makron/McGraw-Hill, 1991.

NITZKE, J.; CARNEIRO, M.; GELLER, M.; SANTAROSA, L. Criação de ambientes de aprendizagem colaborativa. **SBIE´99**, Curitiba, 1999. Disponível em : <http://www.niee.ufrgs.br/~alunospg99/mara>. Acesso em: 20 fev. 2004.

NITZKE, J.; CARNEIRO, M.; GELLER, M. **Aprendizagem cooperativa/colaborativa apoiada por computador (ACAC)**. Disponível em: <http://www.niee.ufrgs.br/~alunospg99/mara>. Acesso em: 10 jan. 2004.

SHAOFENG, W. The role of java RMI in designing workflow management system. **ACM SIGSOFT Software Engineering Notes,** v.26, n.2, p. 49-52, 2001.

SILVA FILHO, R.S.; WAINER, J.; MADEIRA, E.R.M. et al. **WONDER**: a distributed architecture for large scale workflow using CORBA. **17º SBRC – Simpósio Brasileiro de Redes de Computadores.** (2002). Disponível em: <www.dcc.unicamp.br/~931680/wonder>. Acesso em: 10 jan. 2004.

SORENSEN, C. **Why case tools do not support co-ordination**. (1995) Disponível em: <http://is.lse.ac.uk/staff/sorensen/publications.htm>. Acesso em: 10 set. 2003. p. 4/1-4/3.

TANENBAUM, A. S. **Sistemas operacionais modernos**. Porto Alegre: Bookman, p. 247-269, 2000.

TIJIBOY, A. V.; MAÇADA, D. L. **Cooperação e colaboração**: o nosso conceito. Disponível em: <http://www.niee.ufrgs.br/cursos/topicos-ie/ana/conceito.htm>. Acesso em: 12 mar. 2004.

Unified modeling language – UML (2003) Disponível em: <http://www.omg.org/docs/formal/03-03-01.pdf>. Acesso em: 12 jan. 2004.

VERSEY, I.; SRVANAPUDI, P. A. CASE tools as collaborative support technologies. **Communications of ACM**, v.38, n.1, p.83-95, 1995. Disponível em: <http.www.portal.acm.org>. Acesso em: 15 jan. 2004.

XU B; ZHIJUM, HE, X. Y.; Enhancing coordination in global cooperative software design. **Proceedings of the Ninth International Conference on Computer Supported Cooperative Work in Design, 2005, IEEE CNF**, p 22 - 26 Vol. 1. Disponível em: <http://ieeexplore.ieee.org>. Acesso em: 27 out. 2004.

Apêndices

SIMULAÇÃO DO PROTÓTIPO

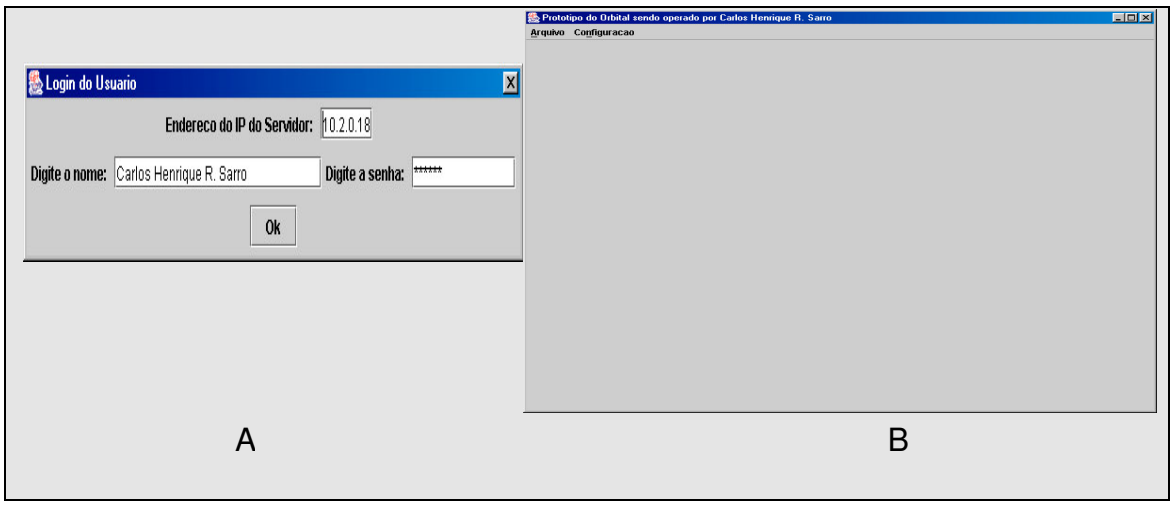


FIGURA 38 - (A) LOGIN DE ENTRADA E IDENTIFICAÇÃO, (B) MENU PRINCIPAL

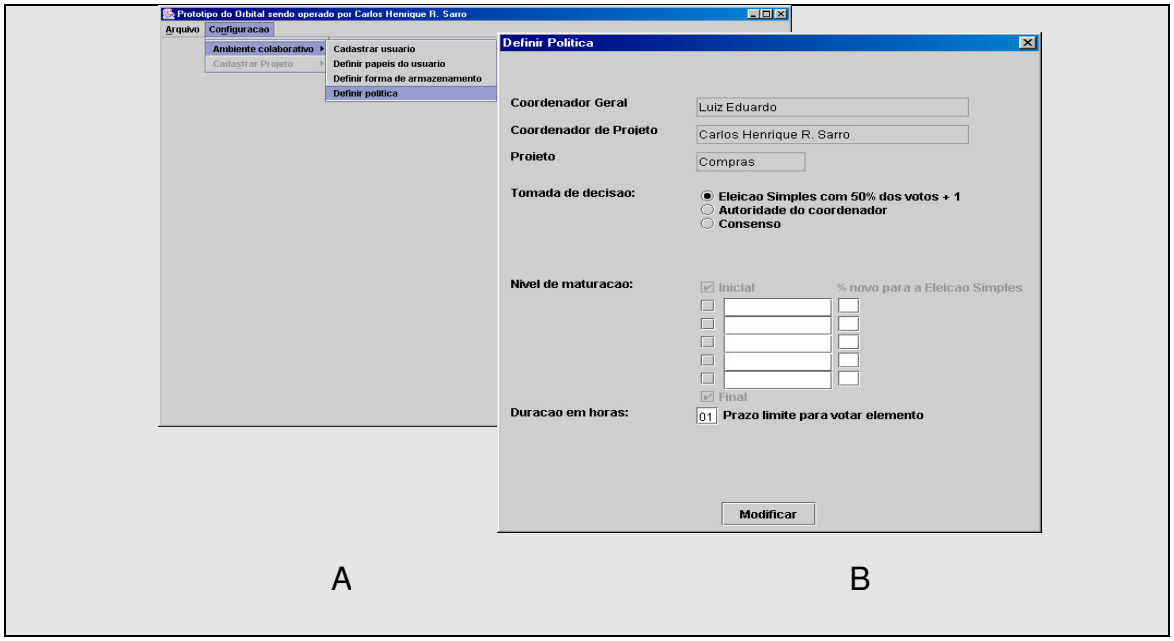


FIGURA 39 - (A) ACESSO AO MENU DE CONFIGURAÇÃO PELO COORDENADOR DE PROJETO, (B) CONFIGURAR O PROJETO PARA TOMADA DE DECISÃO: ELEIÇÃO SIMPLES.

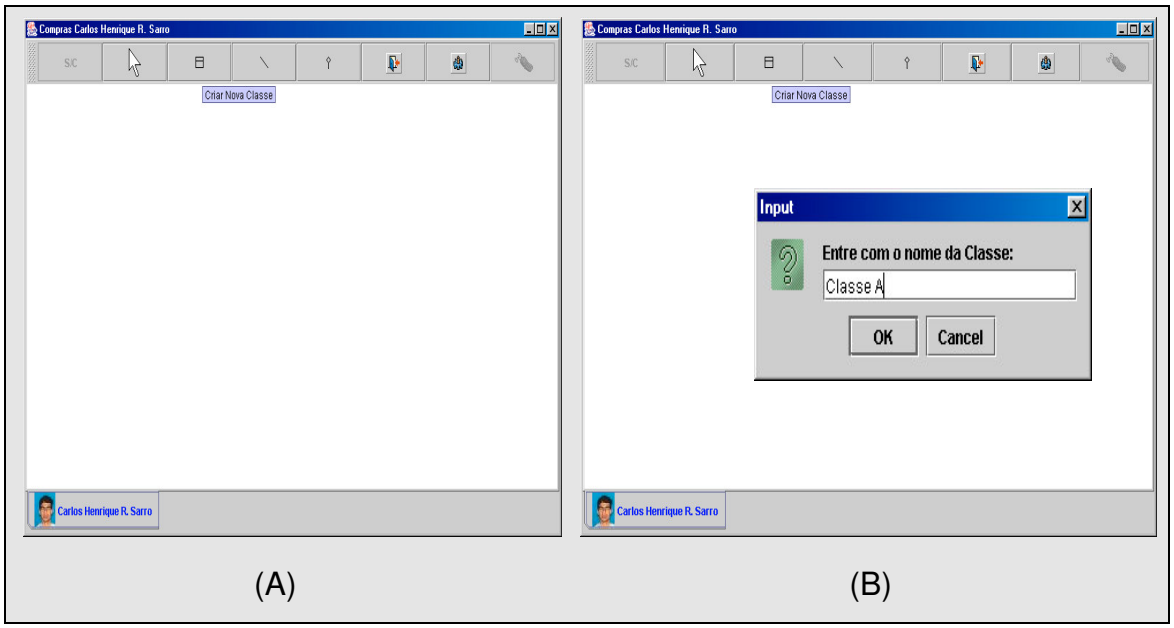


FIGURA 40 - A) INTERFACE DE MODELAGEM COM BARRA DE FERRAMENTA, (B) CRIANDO UMA CLASSE (CLASSE A) NA ÁREA DE TRABALHO LOCAL.

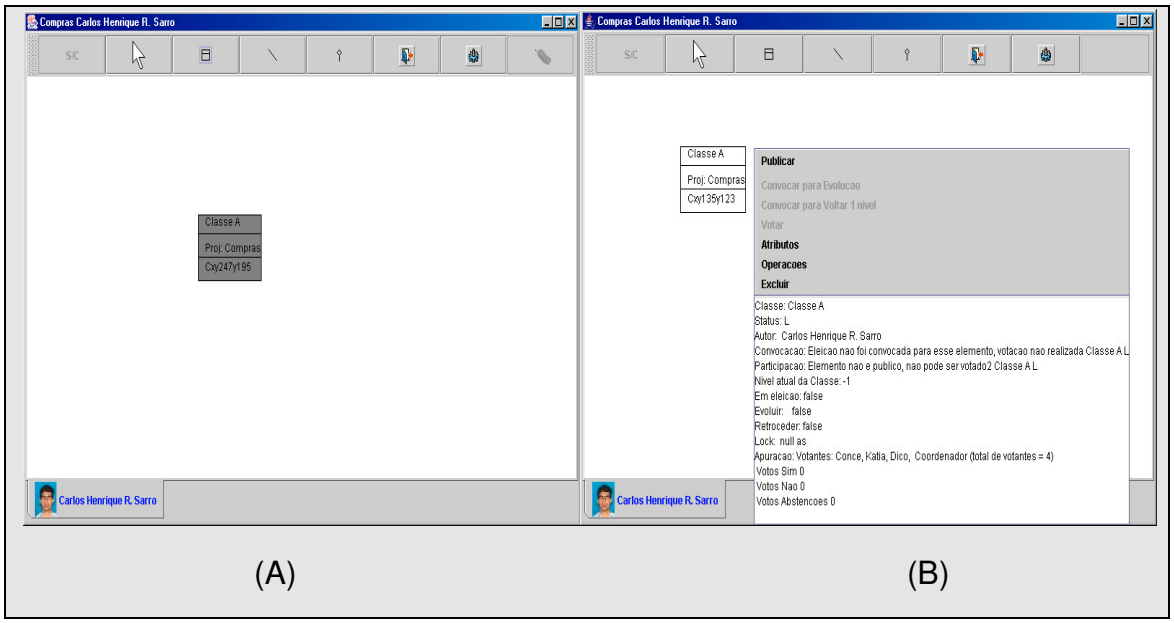


FIGURA 41 - (A) CLASSE CRIADA NA ÁREA DE TRABALHO LOCAL, ESTÁGIO EMBRIONÁRIO, (B) ACESSO AO MENU COM ACIONAMENTO ATRAVÉS DO MOUSE.

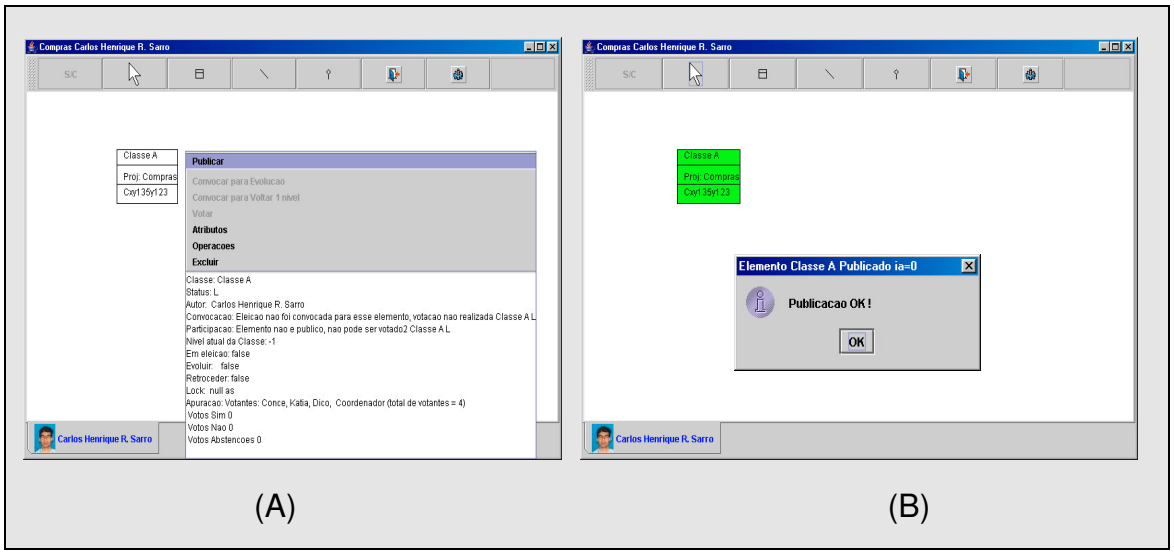


FIGURA 42 - (A) CLASSE A SENDO PUBLICADA, (B) CLASSE B PUBLICADA COM SUCESSO.

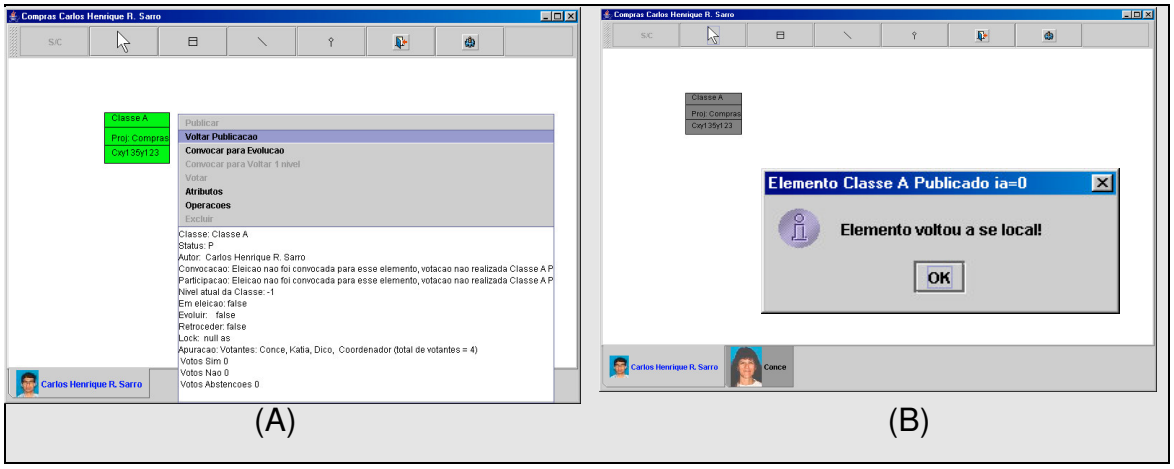


FIGURA 43 - (A) MODELADOR ACIONA RETORNAR PUBLICAÇÃO, (B) CLASSE A VOLTOU NO ESTÁGIO EMBRIONÁRIO, NOVO MODELADOR (CONCE) ENTRA NA SESSÃO DE TRABALHO COLABORATIVO

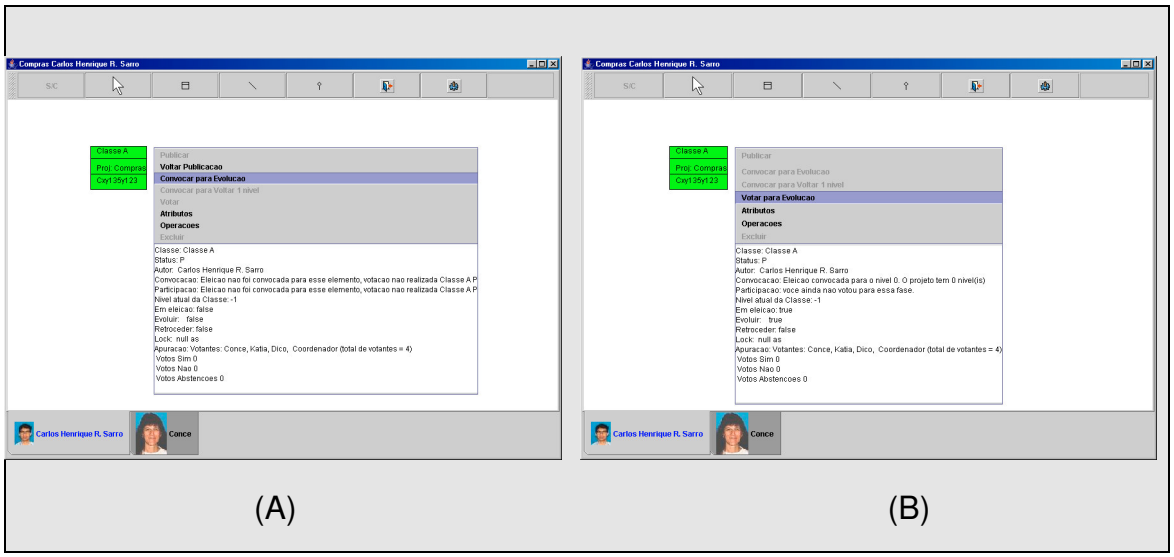


FIGURA 44 - (A) MODELADOR ACIONA A OPÇÃO DE 'CONVOCAR PARA EVOLUIR', (B) OPÇÃO DE 'VOTAR PARA EVOLUIR' FICA DISPONÍVEL PARA VOTAR

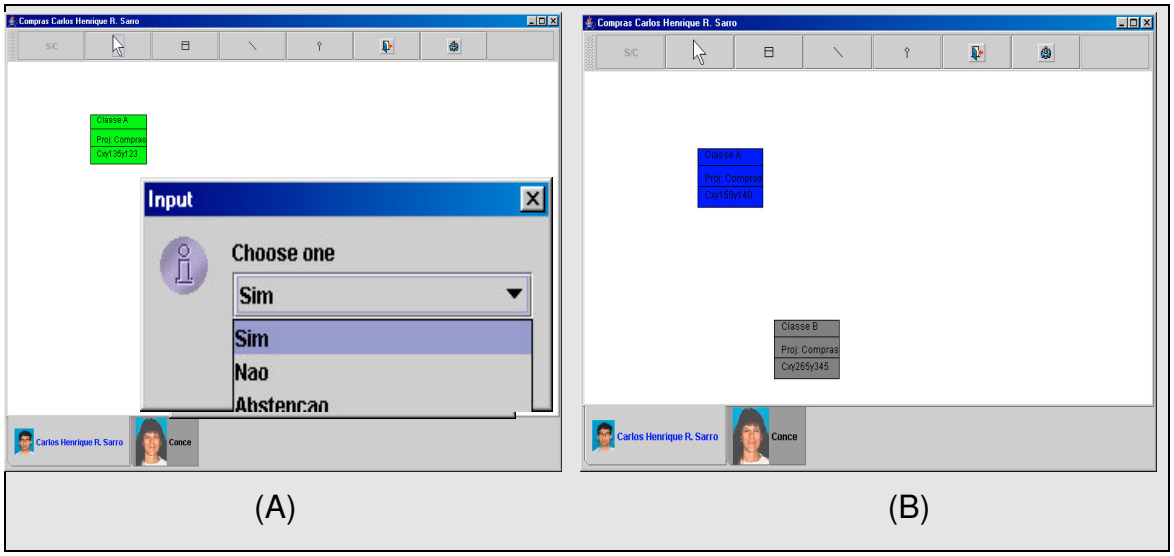


FIGURA 45 - (A) MODELADOR VOTANDO ENTRE AS TRÊS OPÇÃO POSSÍVEIS DE VOTO (SIM, NÃO, ABTENSÃO) PARA A CLASSE EM COR VERDE. (B) VOTAÇÃO CONCLUÍDA, ELEMENTO LIBERADO PARA QUE OUTROS MODELADORES POSSAM VOTAR

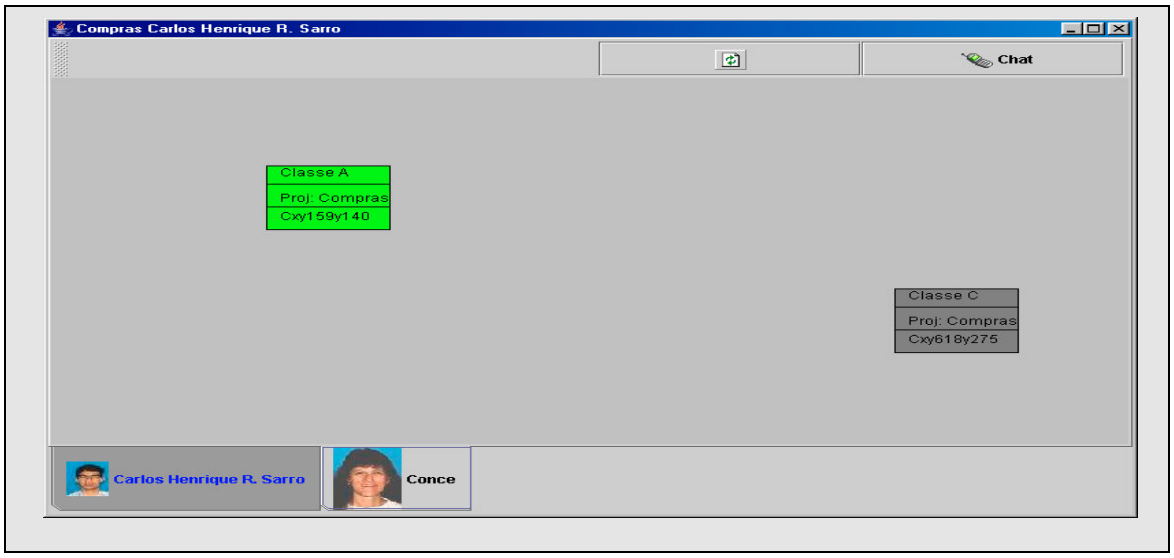


FIGURA 46 - MODELADOR CARLOS VISUALIZANDO A ÁREA DE TRABALHO DA MODELADORA CONCE

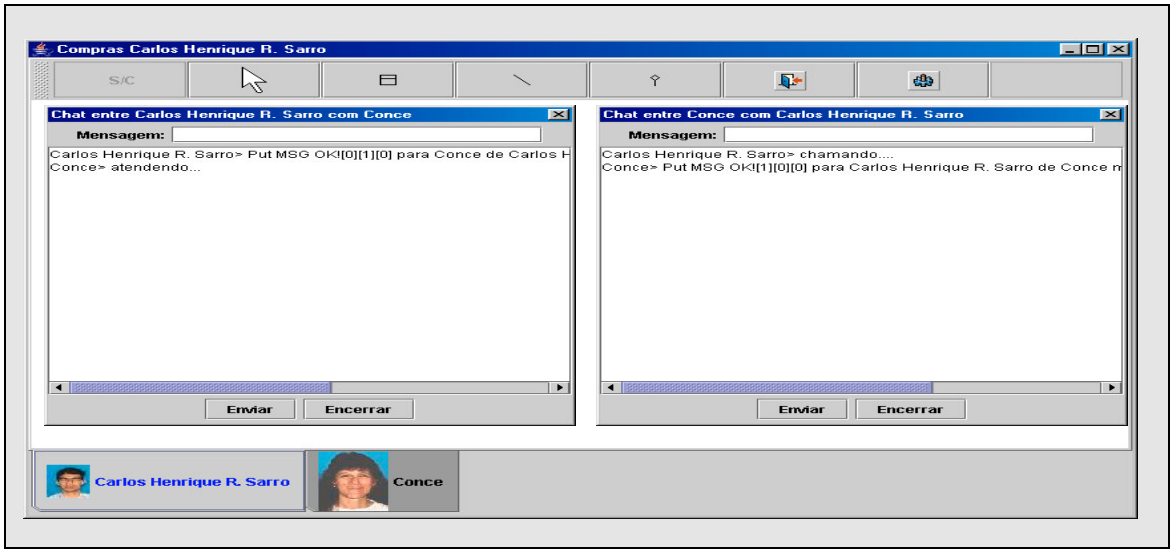


FIGURA 47 - CHAT ENTRE OS MODELADORES CARLOS E CONCE DURANTE A SESSÃO DE TRABALHO COLABORATIVO