UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Câmpus de Presidente Prudente

Leandro Meira Marinho Queiróz

PGCP - Plataforma para Gerenciamento Colaborativo de Projetos

2011

Leandro Meira Marinho Queiróz

PGCP - Plataforma para Gerenciamento Colaborativo de Projetos

Trabalho de conclusão de curso apresentado ao Departamento de Matemática, Estatística e Computação (DMEC), da Universidade Estadual Paulista "Júlio de Mesquita Filho".

Orientador: Prof. Dr. Ronaldo Celso Messias Correa

Co-orientador: Prof. Dr. Rogério Eduardo Garcia

Presidente Prudente

2011

Termo de Aprovação

Leandro Meira Marinho Queiróz

PGCP - Plataforma para Gerenciamento Colaborativo de Projetos

Trabalho de Conclusão de Curso, apresentado por Leandro Meira Marinho Queiróz e aprovado em 20 de Dezembro de 2011, em Presidente Prudente, Estado de São Paulo, pela banca examinadora constituída pelos professores:

Prof. Dr. Ronaldo Celso Messias Correa
Universidade Estadual Paulista

Prof. Dr. Rogério Eduardo Garcia
Universidade Estadual Paulista

Prof. Raphael Garcia
Universidade Estadual Paulista

Presidente Prudente, 20 de Dezembro de 2011

Resumo

Sistemas Colaborativos são projetados para oferecer suporte a indivíduos a fim de resolver um problema em conjunto, ou seja cooperativamente (ou colaborativamente). Tais sistemas possuem a vantagem de não necessitar da presença em um mesmo local dos indivíduos participantes. A base de qualquer Sistema Colaborativo é a comunicação. Os usuários interagem entre si a fim de trocar informações. A comunicação entre os indivíduos é realizada com a utilização de ferramentas colaborativas, tais como: bate-papo, correio eletrônico, repositório de documentos, etc.. Com o uso destas ferramentas, empresas podem diminuir gastos operacionais, economizando com viagens e hospedagem de seus funcionários para presenciar reuniões. Portanto, os integrantes de projetos apenas necessitam de um computador conectado à internet para utilizar as ferramentas colaborativas. Já a gerência de projetos é essencial para a qualidade do produto final. Para realizar um gerenciamento adequado, metodologias devem ser aplicadas utilizando métodos, ferramentas e procedimentos. A estas metodologias dá-se o nome de modelo de processo de software. Existem algumas fases que são consideradas genéricas em todos os modelos, são elas: definição, desenvolvimento, manutenção e atividade de apoio ao processo de software. Este trabalho descreve uma plataforma de gerenciamento de projetos usando os conceitos e funcionalidades dos Sistemas Colaborativos. O objetivo desta plataforma é unificar várias ferramentas colaborativas (como bate-papo e repositório de documentos, audio e videoconferência) em um sistema de gerenciamento de projetos. Utilizando o sistema, torna-se possível o gerenciamento de projetos, além da possibilidade de manter uma comunicação constantemente ativa entre os integrantes.

palavras-chave: Sistemas Colaborativos. Sistemas Cooperativos. Ferramentas Colaborativas. Gerência de Projetos.

Abstract

Collaborative Systems are designed to support individuals in order to solve a problem together, or cooperatively (or collaboratively). Such systems have the advantage of not requiring the presence in the same location of the participants. The basis of any Collaborative System is communication, where users interact with each other in order to exchange information. Communication among individuals is accomplished through collaborative tools such as chat, email, document repository, etc.. Using these tools, companies can reduce operational costs, saving on travel and accommodation of its employees to attend meetings. The project members only need a computer with internet access to use collaborative tools. Project management is essential to the quality of the final product. To perform a proper management, methodologies must be applied using methods, tools and procedures. There are some phases that are considered generic for all software process models, they are: definition, development, maintenance and activity to support the software process. In this paper is described a platform for project management using the concepts and features of Collaborative Systems. The goal of this platform is to unify various collaborative tools (such as chat and document repository) in a project management system. Using the system, it becomes possible to manage projects, and the ability to maintain an active constant communication between members of each project.

keywords: Collaborative Systems. Cooperative Systems. Communication. Collaborative Tools. Project management.

Sumário

1 Introdução p. 9

1.1 Objetivos . . . . . p. 10

1.2 Organização do Trabalho . . . . . p. 10

2 Revisão Bibliográfica p. 11

2.1 Gerenciamento de Projetos . . . . . p. 11

2.2 Ambientes Colaborativos . . . . . p. 13

2.3 Ferramentas Utilizadas . . . . . p. 18

2.3.1 Linguagem Java . . . . . p. 18

2.3.2 NetBeans . . . . . p. 20

2.3.3 SGBD MySql . . . . . p. 22

2.3.4 Protocolos de Rede . . . . . p. 22

2.4 Plataformas Existentes . . . . . p. 23

2.4.1 Planner . . . . . p. 23

2.4.2 OpenProj . . . . . p. 24

2.4.3 Microsoft Project . . . . . p. 25

2.4.4 Basecamp . . . . . p. 25

3 PGCP - Plataforma para Gerenciamento Colaborativo de Projetos p. 28

3.1 Desenvolvimento do sistema . . . . . p. 29
3.1.1 Estrutura . . . . . p. 32
3.1.1.1 Autenticação . . . . . p. 32
3.1.1.2 Gerenciar Usuários . . . . . p. 37
3.1.1.3 Correio Eletrônico . . . . . p. 38
3.1.1.4 Operações sobre Projeto . . . . . p. 40
3.1.2 Bate-papo . . . . . p. 46
3.1.3 Repositório de Arquivos . . . . . p. 50
3.1.4 Áudio e Videoconferência . . . . . p. 51

4 Estudo de Caso - Sistema Proex p. 57
4.1 Introdução . . . . . p. 57
4.2 Sistema PROEX na PGCP . . . . . p. 57

5 Conclusão p. 70

Referências p. 72

Apêndice A – Documento de Requisitos p. 74

Lista de Figuras

1 Elementos dos Ambientes Colaborativos . . . . . p. 15

2 Elementos-Chave dos Ambientes Colaborativos . . . . . p. 16

3 Interface - NetBeans . . . . . p. 21

4 Interface - Planner . . . . . p. 24

5 Interface - OpenProj . . . . . p. 25

6 Interface - Microsoft Project . . . . . p. 26

7 Interface - BaseCamp . . . . . p. 27

8 Diagrama de Casos de Uso. . . . . p. 31

9 Diagrama de Classes - PGCP Cliente . . . . . p. 33

10 Diagrama de Classes - PGCP Servidor . . . . . p. 34

11 Modelagem da Base de Dados . . . . . p. 35

12 DSS Efetuar Login . . . . . p. 37

13 DSS Enviar Mensagem . . . . . p. 39

14 Diagrama de Colaboração - Enviar Mensagem . . . . . p. 39

15 DSS Abrir Projeto . . . . . p. 41

16 DSS Adicinar Recurso a Projeto . . . . . p. 42

17 DSS Atribuir Usuário como Administrador de Projeto . . . . . p. 43

18 DSS Adicionar Recursos a uma Fase . . . . . p. 44

19 DSS Adicionar Atividade a uma Fase . . . . . p. 45

20 DSS Adicionar um Comentário a uma Fase . . . . . p. 46
21 DSS Iniciar chat em uma Fase . . . . . p. 49
22 Diagrama de Colaboração - Iniciar chat em uma Fase . . . . . p. 50
23 DSS Upload Arquivo em uma Fase . . . . . p. 52
24 Diagrama de Colaboração - Upload Arquivo em uma Fase . . . . . p. 52
25 DSS Download Arquivo em uma Fase . . . . . p. 53
26 DSS Iniciar Audioconferência em uma Fase . . . . . p. 55
27 Diagrama de Colaboração - Iniciar Audioconferência em uma Fase . . . p. 55
28 DSS Iniciar Videoconferência em uma Fase . . . . . p. 56
29 Fluxo de operações do usuário. . . . . p. 58
30 Interface Principal. . . . . p. 59
31 Interface de alteração de Projeto. . . . . p. 60
32 Interface principal de Projeto. . . . . p. 60
33 Interface Recursos do Projeto. . . . . p. 61
34 Interface Adicionar Recursos do Projeto. . . . . p. 62
35 Interface Fases. . . . . p. 63
36 Interface Adicionar Fase. . . . . p. 63
37 Interface Adicionar Recursos a uma Fase. . . . . p. 64
38 Interface Gerenciar Atividades da Fase. . . . . p. 64
39 Interface Gerenciar Comentários da Fase. . . . . p. 65
40 Interface Gerenciar Chats da Fase. . . . . p. 66
41 Interface Chat. . . . . p. 66
42 Interface Gerenciar Repositório de Documentos da Fase. . . . . p. 67
43 Interface Gerenciar Audioconferências da Fase. . . . . p. 67
44 Interface Transmissão de Audioconferência. . . . . p. 68
45 Interface Meu Correio. . . . . p. 68

Capítulo 1

Introdução

Inicialmente são apresentados dois conceitos-chaves: a gerência de projetos e os ambientes colaborativos.

Como o foco da plataforma desenvolvida é a gerência de projetos de softwares, pode-se basear-se em conceitos da Engenharia de Software para explicar a gerência de projetos, embora com o uso da plataforma desenvolvida seja possível realizar a gerência de qualquer tipo de projeto baseado em fases.

A gerência de projetos (ou gestão de projetos) é definida por um conjunto de processos ou ações que visam garantir, basicamente, o sucesso no desenvolvimento de um projeto. O sucesso é alcançado, ou não, levando-se em conta fatores como: qualidade do produto final, cumprimento de prazos, orçamento e requisitos, dentre outros. [1] Existem inúmeros softwares no mercado que oferecem os mais diversos recursos para a gerência de projetos, os quais são apresentados na seção 2.4. Tais softwares auxiliam o gerente de projetos, podendo acompanhar o desenvolvimento de fases ou ainda ter uma visão geral do projeto.

Já os ambientes colaborativos têm como foco a comunicação. Eles também são conhecidos como ambientes cooperativos, sistemas colaborativos, sistemas cooperativos, dentre outras variantes (neste trabalho é utilizado o termo ambientes colaborativos). Ambientes colaborativos têm como objetivo oferecer ferramentas (como bate-papo, correio eletrônico, repositório de documentos, áudio e videoconferência) pelas quais indivíduos podem discutir problemas em grupo, sem a necessidade de estar no mesmo local. A grande vantagem dos ambientes colaborativos é o trabalho remoto, minimizando as barreiras de tempo e espaço e tornando o trabalho em equipe mais eficiente.

Os detalhes de gerência de projetos e ambientes colaborativos são abordados no de-

correr deste texto.

1.1 Objetivos

A plataforma desenvolvida, intitulada como "PGCP - Plataforma para Gerenciamento Colaborativo de Projetos", realiza uma unificação da gerência de projetos e dos ambientes colaborativos, criando um sistema de gerência de projetos, porém juntamente com diversas ferramentas colaborativas. Desta forma, a plataforma dá suporte tanto a gerentes de projetos quanto a participantes do projeto, oferecendo, além de ferramentas de gerência de projetos, ferramentas colaborativas que permitem a todos realizar reuniões virtuais de diferentes formas, além da possibilidade dos usuários realizarem comentários sobre fases, compartilhar arquivos com os demais participantes, dentre alguns outros recursos.

Os Ambientes Colaborativos têm como objetivo tentar minimizar as barreiras de tempo e espaço, ou seja, permitir o trabalho em equipe de modo mais fácil e eficaz. Eles devem ser especializados o bastante para oferecer as ferramentas adequadas aos usuários afim de facilitar a comunicação, a coordenação e o controle entre as partes. A PGCP focaliza a gerência das fases do projeto de software e das atividades nelas desenvolvidas, oferecendo suporte a acompanhamento de cumprimentos de prazos e cronogramação.[2]

1.2 Organização do Trabalho

NO Capítulo 2 há descrições e detalhes sobre ambientes colaborativos, gerenciamento de projetos, ferramentas utilizadas na implementação e algumas plataformas existentes.

No Capítulo 3 é descrita a PGCP, apresentando diagramas da Engenharia de Software e detalhes da implementação do sistema.

No Capítulo 4 é apresentado um estudo de caso realizado, introduzindo a PGCP para auxílio do gerenciamento do sistema da PROEX.

No Capítulo 5 são apresentados os resultados e conclusões.

Por fim, são apresentadas as Referências Bibliográficas e o Apêndice.

Capítulo 2

Revisão Bibliográfica

2.1 Gerenciamento de Projetos

Nesta seção são abordados conceitos sobre a gerência de projetos e os fatores envolvidos, desenvolvendo uma descrição sobre os principais componentes do processo de software.

O planejamento de todo e qualquer projeto implementado por uma organização é essencial para a qualidade do produto final. No universo de um projeto, existe o *usuário*, o qual contrata uma *organização*, que possui *desenvolvedores*.

No desenvolvimento de um projeto, a organização possui uma maior quantidade de interesses, pois nela estão incluídos os interesses dela própria (como cumprimento de prazos), além dos interesses do desenvolvedor (como facilidade de manutenção e cumprimento dos requisitos) e do usuário (facilidade de uso, desempenho, preço). O desenvolvedor possui um papel intermediário, levando em conta seus próprios interesses, além dos interesses do usuário. O usuário, por sua vez, não necessita saber as políticas e ferramentas que a organização ou o desenvolvedor estão utilizando, tendo em vista apenas seus próprios interesses. Existem diversas pessoas, com diferentes funções, envolvidas no processo de software. Exemplos são: Analistas, consultores, documentadores, programadores, gerentes, gerentes de banco de dados, etc..

O cliente, normalmente, é apenas um grupo de pessoas que possue a ideia do software. Na maioria das vezes, a maneira como o cliente descreve o software ao gerente de projeto, não é clara. A partir daí, o gerente descreve o projeto à equipe e cada integrante pode visualizar o software de maneira diferente. Isto não é agradável nem à empresa e nem

ao cliente, pois corre-se o risco do software não ser como o cliente havia pensado, e para agravar ainda mais, podem ocorrer atrasos por falta de planejamento, além do custo do software poder variar, devido à diferença nas ideias de como o software deveria ser.

Desta forma, são apresentados alguns detalhes e conceitos sobre o processo de software, destacando os principais itens a ser abordados no gerenciamento de um projeto de software. O processo de software abrange um conjunto de três elementos fundamentais: métodos, ferramentas e procedimentos.[1]

Para a realização de projetos de software, é essencial a aplicação de metodologias que estabelecem uma ordem na qual as fases de um projeto devem ser realizadas, utilizando os métodos, ferramentas e procedimentos descritos acima. A aplicação de tais metodologias é referida como modelo de processo de software. Segundo Pressman [1]: "Um modelo de processo para engenharia de software é escolhido com base na natureza do projeto e da aplicação, nos métodos e ferramentas a serem usados, e nos controles e nos produtos intermediários e finais que são requeridos." Existem diversos modelos de processo de software. Entre os principais estão: modelo sequencial linear, modelo de prototipagem, modelo RAD (desenvolvimento rápido de aplicação), modelo incremental, modelo espiral, modelo de desenvolvimento concorrente, modelo de montagem de componentes e modelos de métodos formais.

Apesar de haver diversos tipos de modelos de processo de software, existem algumas fases genéricas que geralmente estão presentes em todos eles. São elas:[1]

- Fase de definição: nesta fase ocorre a definição do projeto, ao passo que o cliente deve especificar os requisitos do sistema, montando um documento que explique o que o sistema deve e o que não deve fazer. Esta fase também tem como objetivo realizar o planejamento do projeto, ou seja, ela focaliza "o que"será desenvolvido. Nela estão contidas 3 tarefas principais: engenharia de sistemas, planejamento do projeto de software e análise de requisitos. No apêndice deste trabalho está a especificação de requisitos do sistema implementado.

- Fase de desenvolvimento: é nesta fase onde ocorre o desenvolvimento do projeto. Todo o precesso de projeto de software é realizado aqui, além da implementação, posteriorizada pelas rotinas de teste, para assegurar que o software atendeu os requisitos e é de qualidade. Esta fase tem foco em "como"o software será desenvolvido. Tarefas principais: projeto de software, geração de código e inspeção e teste de software.

- Fase de manutençao: é comum pensar que uma vez que o software esteja terminado, a única coisa a se fazer é entregá-lo ao cliente. Porém, erros que passaram despercebidos podem ocorrer, os quais só serão encontrados na utilização do software. Além disso, o ambiente no qual o software é utilizado pode sofrer alterações, exigindo atualizações. Ou seja, esta fase focaliza nas "mudanças"que possam ocorrer no software. As principais mudanças são: correção de erros/defeitos, adaptações exigidas conforme o ambiente do software evolui e mudanças devido a melhoramentos ocorridos por alterações nos requisitos do cliente.

- Atividades de apoio ao processo de software: durante todo o processo de software, há algumas atividades que complementam e podem ser realizadas em todas as fases, a fim de minimizar ainda mais os erros. Algumas atividade comuns são: Revisões técnicas formais, garantia de qualidade de software, gerenciamento de configuração de software.

As fases descritas acima podem conter inúmeras outras tarefas, dentre as quais é válido mencionar algumas importantes: Análise de riscos, revisões, métricas e estimativas e análise de qualidade de software. Porém, devido ao fato da PGCP ter o seu foco na gerência das fases do projeto de software, oferecendo suporte a acompanhamento de cumprimento de prazos, a cronogramação deve ser ressaltada. Realizar um cronograma é importante pelo fato de que em projetos grandes e complexos, sem cronograma, a visualização de todas as fases é praticamente impossível, além de que geralmente mais de uma fase ocorre em paralelo, sendo possível que o resultado de uma fase interfira diretamente sobre outra.[1]

Segundo Pressman [1], um cronograma de um projeto exige que: todas as tarefas estejam presentes na rede, o esforço e a duração devem ser inteligentemente atribuídos a cada tarefa e marcos de referência pequenos (tentar dividir fases grandes em fases menores) sejam estabelecidos a fim de um melhor acompanhamento do progresso.

2.2 Ambientes Colaborativos

No século passado, tornou-se possível armazenar informação não apenas no papel, mas também em fitas magnéticas. Este tipo de armazenamento talvez seja responsável por abrir as portas para a evolução do tipos de armazenamento de informação. Atualmente, um simples conjunto de caixas de som e microfone faz com que seja possível uma única pessoa falar a milhares. Pode-se imaginar como era extremamente difícil os generais da

antiguidade falar a vinte mil homens, por exemplo. Hoje em dia qualquer um é capaz de registrar algo em áudio ou vídeo, ou seja, fica evidente que os meios de comunicação estão em constante evolução. A quantidade de meios de comunicação aumentou significantemente. Atualmente, é possível comunicar-se através da televisão, rádio, e-mail, mensageiros instantâneos, áudio e videoconferência, etc. [3]

Hoje, vive-se no que pode ser chamada de Era da Informação e da Comunicação. Uma realidade inédita, onde o principal trabalho do homem é ser criativo e ter ideias. O restante fica por conta das máquinas. A comunicação só pode ser realizada por duas ou mais pessoa. Pensando num ambiente corporativo, sabe-se que a comunicação é crucial para o desenvolvimento e sucesso da empresa.

Com o passar dos séculos, os homens moveram o trabalho, que antes era realizado em casa, para a empresa. Hoje, o que está acontecendo é justamente o contrário. Cada vez mais há funcionários abandonando o ambiente empresarial e prestando seus serviços em suas casas. Também é muito comum funcionários prestarem serviços entre as filiais de empresas remotamente. Enfim, é muito comum que o funcionário trabalhe para a empresa sem estar presente nela. [3]

Empresas possuem funcionários e projetos. Os funcionários devem se organizar com o propósito de concluir o projeto da melhor maneira possível. Porém, os que trabalham presencialmente nas empresas devem interagir com os que trabalham remotamente. Outro ponto importante é a necessidade de que, muitas vezes, a troca de informações entre os funcionários deve ser realizada em tempo real.

Gestão de Conhecimento é a busca pela melhora do desempenho das empresas utilizando condições empresariais favoráveis, processos de localização, ferramentas e tecnologias de informação e comunicação, partilha e criação do conhecimento. Uma boa prática de Gestão do Conhecimento influencia diretamente no desenvolvimento da empresa. Entende-se por conhecimento a informação interpretada, ou seja, o significado de cada informação e suas implicações. [3]

Num certo momento, as empresas viram-se numa posição de necessidade da criação de um ambiente onde todos pudessem se comunicar adequadamente, oferecendo todas as ferramentas necessárias para a troca de informação, com o intuito de aproximar os indivíduos que se encontravam em posições geográficas distintas. Deu-se então o surgimento do que hoje é chamado de Ambientes Colaborativos. Segundo Carla Oliveira [2], Ambientes Colaborativos são sistemas utilizados em rede, projetados para oferecer suporte a indivíduos que desejam resolver um problema em cooperação com outros (em grupo), sem

que seja necessária a presença de todos os indivíduos no mesmo local. A base de qualquer Ambiente Colaborativo é a comunicação, onde usuários interagem entre si afim de trocar informações sobre determinado assunto.

As principais finalidades dos Ambientes Colaborativos são as seguintes: [2]

- Gerenciamento e coordenação do trabalho em equipe;
- Integração dos trabalho em todos os níveis e funções;
- Integração da organização com o meio externo, como: clientes, distribuidores, fornecedores, organizações governamentais e não governamentais;
- Gerenciamento, criação, armazenamento, recuperação e compartilhamento de documentos;
- Definição da programação de tarefas para indivíduos e grupos;
- Facilitar a comunicação e o gerenciamento de contatos e relacionamentos entre os indivíduos internos e externos a empresa.

Tais elementos dos Ambientes Colaborativos podem ser representados pela Figura 1[4].

Figura 1: Elementos dos Ambientes Colaborativos

Ambientes Colaborativos fazem parte de um universo maior, o dos Sistemas de Informação, os quais são praticamente imprescindíveis para qualquer organização (de médio

ou grande porte) pelo fato destes possuírem funções fundamentais. Dentre tais funções, pode-se citar: suporte aos processos e operações, apoio à tomada de decisão por gerentes e funcionários de todos os níveis hierárquicos e suporte às estratégias competitivas da empresa. É fácil notar que para a realização de todas estas funções pode ser necessário o trabalho em equipe, o que acabou resultando numa grande adoção dos Ambientes Colaborativos pelas empresas, dado seu ótimo desempenho, que juntamente com o uso consciente, proporcionam resultados bastante satisfatórios. Dá-se o nome de Ambientes Colaborativos ou Cooperativos aos Sistemas de Informação com foco específico no trabalho em equipe. [5]

Segundo o Dicionário Aurélio, colaborar significa "Trabalhar com uma ou muitas pessoas numa obra (...); cooperar (...)". Na primeira etapa do processo de colaboração tem-se a comunicação. A partir desta, passam a ocorrer negociações as quais levam à resolução de um problema ou à conclusão de um trabalho. Um termo bastante utilizado para se referir aos Ambientes Colaborativos é o Groupware, palavra composta da junção das palavras group (grupo) e software. Há várias outras formas de fazer referência aos Ambientes Colaborativos, como: Sistemas Workflow, CSCW (Computer Supported Cooperative Work), Online Collaboration, Web Collaboration, Colaboração online, Collaboration tools, Colaboração via web, Sistema de Colaboração, Sistema Colaborativo, etc. [2]

Como pode-se perceber, Ambientes Colaborativos dependem, resumidamente, de três elementos-chave: Cooperação, Comunicação e Coordenação. Eles, as suas etimologias e relações são representados na Figura 2[4].

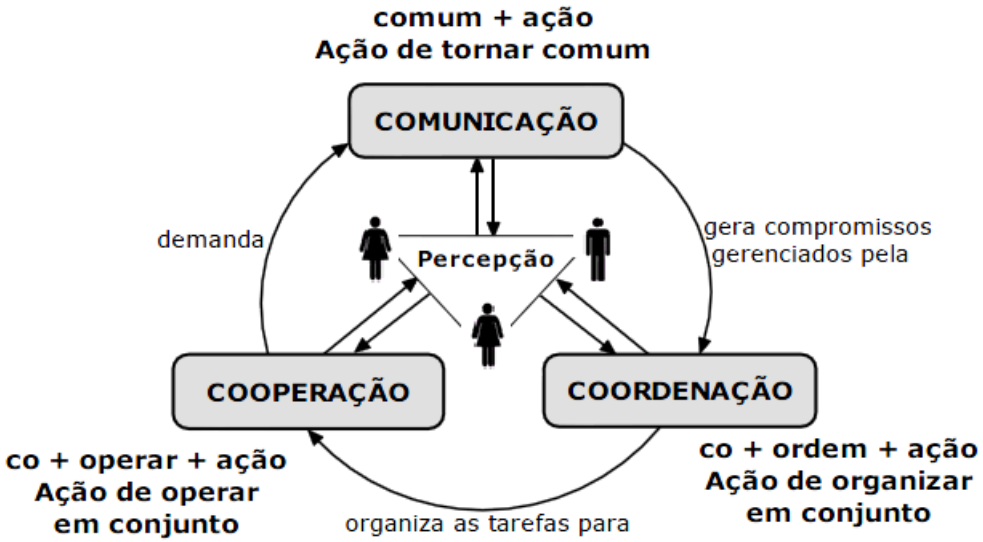

Figura 2: Elementos-Chave dos Ambientes Colaborativos

Na Figura 2, pode-se perceber a geração de um ciclo em torno dos três elementos. Partindo da Comunicação, esta gera compromissos que devem ser gerenciados pela Coordenação. Esta, por sua vez, tem o papel de organizar as tarefas para que a Cooperação seja bem sucessida e produtiva. A Cooperação, por fim, claramente demanda Comunicação, pois é impossível cooperar sem comunicar.

Os Ambientes Colaborativos podem ser classificados conforme o lugar das interações (presenciais ou à distância) e o tempo (síncronas e assíncronas). Considerando o fator tempo, ferramentas síncronas necessitam de um tempo de resposta imediato ou curto, ou seja, devem ser executadas em tempo real (Mensageiros Instantâneos, audio e videoconferências). As ferramentas assíncronas, por sua vez, não requerem tempo de resposta imediato (e-mails, fóruns). [2] Segundo Carmargo [6], Ambientes Colaborativos podem ser compostos por diversas funcionalidades e ferramentas, algumas delas descritas abaixo:

- Agenda: Permite a elaboração de agendas individuais, para um grupo ou para a organização. Através dela é possível reservar horários e recursos (salas, equipamentos). Armazena informações, como telefone e e-mail, sobre todos os indivíduos relacionados ao projeto, inclusive de indivíduos e organizações externas;

- Repositório de documentos: Documentos são armazenados pelos usuários do sistema, para uso geral ou de um grupo específico (apenas coordenadores, por exemplo). O repositório pode oferecer controle de versão;

- Áudio e videoconferência: São recursos síncronos, ou seja, que requerem resposta em tempo real. São usados para estabelecer contato entre indivíduos ou grupos. A áudioconferência pode ser realizada, por exemplo, através de voz sobre IP (VOIP). A videoconferência une áudio e vídeo em uma transmissão sincronizada;

- Reuniões Virtuais: Através dos recursos de áudio e videoconferência e bate-papo, faz-se possível a realização de reuniões entre grupos de usuários do sistema;

- Suporte a decisão: Usando as ferramentas que o sistema oferece, é possível proporcionar agilidade na tomada de decisões, utilizando enquetes, votações, etc;

- Fóruns de discussão: Oferece uma estrutura em árvore, dividas por tópicos e subtópicos, possibilitando a discussão de determinados assuntos por grupos de usuários.

- Bate papo: Permite a troca de mensagens instantâneas (de forma síncrona) entre usuários ou grupos. Trata-se de uma solução eficiente e de baixo custo, tornando-se um recurso vastamente usado nos Ambientes Colaborativos.

- Correio Eletrônico: Recurso bastante usado em organizações em geral, o qual tem por objetivo permitir uma comunicação assíncrona entre grupos ou indivíduos. Este é um recurso que deve ser usado de forma controlada devido ao risco de recebimento de mensagens indesejadas (spams);
- Co-autoria de documentos: Possibilita a edição de documentos por mais de um usuário, utilizando um controle de versões, que pode ser feito de várias formas. Este recurso trabalha lado a lado com o Repositório de documentos;
- Fluxo de Trabalho (WorkFlow): Utilizado para controlar e gerenciar o fluxo de trabalho;
- Geradores de Formulários: Permitem a criação de formulários, que padroniza e facilita o fornecimento e compartilhamento de informações e relatórios, não sendo necessário que cada usuário crie um novo documento, e sim preencha um formulário preestabelecido.

2.3 Ferramentas Utilizadas

Nesta seção, são descritas as ferramentas que foram utilizadas na implementação da PGCP. Cada ferramenta ou tecnologia será descrita de forma sucinta. Há também, para cada uma delas, uma justificativa para a sua utilização neste projeto.

As ferramentas e tecnologias usadas são as seguintes:

- Linguagem Java
- NetBeans
- SGBD MySql
- Protocolos de rede

2.3.1 Linguagem Java

Java é uma linguagem de programação criada em 1992 pela empresa Sun Microsystems (empresa hoje pertencente à Oracle). Ela foi desenvolvida para a criação de um controle remoto touchscreen para televisores. O controle remoto se perdeu no tempo, porém a linguagem Java sobreviveu e cresceu em proporções exponenciais ao longo dos anos graças

ao surgimento da internet. Java já havia sido planejada para se "locomover"em redes e sistemas heterogêneos.[7]

Uma das principais características de Java é a portabilidade, pois um mesmo código pode rodar em qualquer plataforma (sistema operacional). Esta portabilidade faz-se possível pois o código escrito em Java é interpretado por uma máquina virtual chamada JVM (*Java Virtual Machine*). Cada sistema operacional (Linux, Mac Os, Windows, Solaris) comunica-se de forma diferente com a JVM, portanto é necessário que haja um tipo de JVM para cada sistema operacional. Porém, uma vez que a JVM esteja instalada no sistema operacional, qualquer código que utilize as bibliotecas padrão do Java pode ser executado independente do sistema operacional.[7]

Para executar um programa em Java, basta que o computador tenha instalado o JRE (*Java Runtime Environment*), o qual possui a JVM e bibliotecas padrão do Java. Atualmente diversos jogos online, programas e plugins necessitam do JRE instalado, o que o torna quase sempre necessário em qualquer computador.[7]

Para desenvolver um programa em Java e ser capaz de executá-lo, é necessário ter instalado o kit de desenvolvimento Java, o JDK (*Java Development Kit*). É através do JDK que o código Java é compilado. Todas as plataformas (sistemas operacionais) para as quais existe JVM também existe JDK, ou seja, é possível desenvolver uma aplicação em Java em Linux, Mac OS, Windows e Solaris.[7]

Para a implementação da PGCP foi utilizado o J2SE (*Java 2 Standard Edition*), que é a versão padrão de desenvolvimento do Java. Existem outras versões, como o J2EE (*Java 2 Enterprise Edition*) e o J2ME (*Java 2 Micro Edition*).[7]

Existem diversos Frameworks que oferecem auxílio ou extensão para o Java. Para a implementaçao da PGCP, foi usado a JMF (*Java Media Framework*). Segundo Daniel G. Costa [7]: "A finalidade dessa API está ligada diretamente à utilização das mídias de áudio e vídeo. Captura, processamento, armazenamento e transmissão em tempo real dessas mídias são os principais serviços oferecidos pela JMF."Através dela, é possível transmitir arquivos de áudio e vídeo, ou até mesmo realizar a captura de áudio ne vídeo no microfone e câmera e transmiti-los em tempo real. A JMF utiliza um protocolo de rede chamado de RTP (*Real time protocol*) para efetuar a comunicação entre computadores. Este protocolo será melhor explicado na subseção Protocolos de Rede. Na PGCP estão presentes os recursos de áudio e videoconferência.[7]

Como descrito no início da subseção, Java oferece suporte à aplicações em rede, através

de *sockets*. Com sua utilização, é possível criar aplicações cliente/servidor, sendo possível a comunicação entre computadores. Não necessariamente um computador é sempre servidor e o outro cliente. Um servidor, em uma determinada comunicação, pode enviar dados a um cliente, e numa próxima comunicação, este servidor pode se transformar em cliente e passar a receber dados do computador que anteriormente era o cliente, e este torna-se o servidor. Java oferece suporte nativo à programação em *sockets* por TCP (*Transmission Control Protocol*) ou por UDP (*User Datagram Protocol*). Na PGCP, *socket* foi implementado através do TCP. A diferença entre tais protocolos e a justificativa da escolha do TCP é apresentada na subseção desta seção chamada Protocolos de Rede.

2.3.2 NetBeans

Existem diversos ambientes onde a programação Java pode ser escrita. A estes ambientes de programação, dá-se o nome de IDE (*Integrated Development Environment* ou Ambiente Integrado de desenvolvimento). As duas IDEs mais utilizadas para a programação em Java é o NetBeans e o Eclipse. Para a implementação da PGCP, o NetBeans foi a IDE escolhida, pois se trata de uma plataforma que atualmente é desenvolvida pela própria Sun Microsystems (atual Oracle), criadora do Java.

O Netbeans, assim como o JRE e o JDK, é compatível com os sistemas operacionais Linux, Mac OS, Windows e Solaris, e ele mesmo é escrito em Java.

As principais características do NetBeans são:

- API Swing desenha componentes como botões e menus, gerando códigos automáticos, auxiliando o programador na criação da interface.
- Plugins para UML (*Unified Modeling Language* ou linguagem de modelagem unificada) e CVS (*Concurrent Version System* ou sistema de versões concorrente).
- Suporte a aplicações web, CSS, banco de dados e servidores web.
- Auto-completar, auxiliando o desenvolvedor na criação de métodos e chamadas a eles, por exemplo.

Na Figura 3 é ilustrada a interface padrão do NetBeans[8].

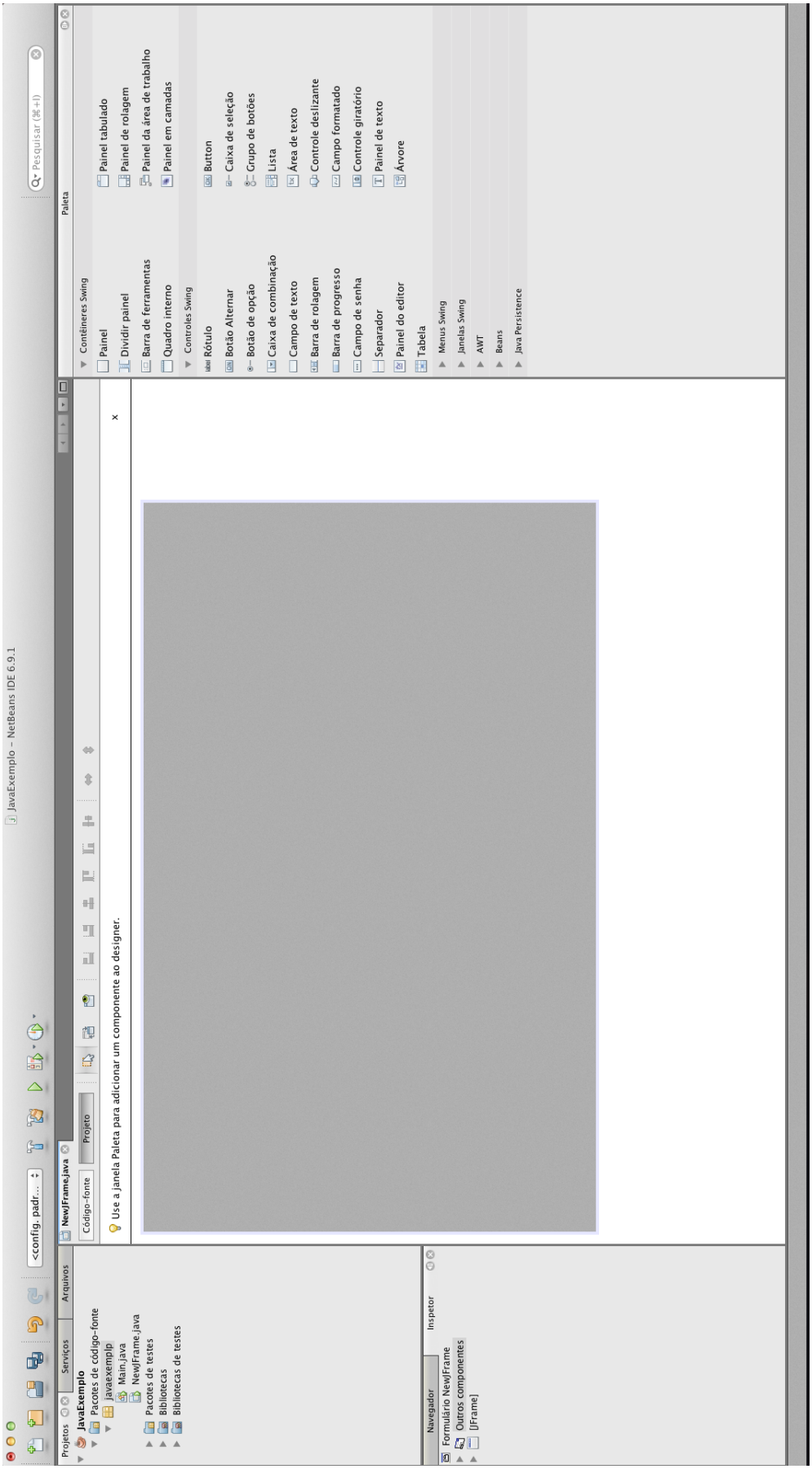

Figura 3: Interface - NetBeans

2.3.3 SGBD MySql

O MySql, SGBD (Sistema Gerenciador de Banco de Dados) usado neste projeto, utiliza a linguagem SQL (*Structured Query Language*), uma linguagem considerada praticamente como padrão na definição de banco de dados. O MySql foi criado na Suécia na década de 90, e atualmente é um dos SGBDs mais usados pelo mundo. Em 2008, a Sun Microsystems (atual Oracle) comprou os direitos sobre o MySql. Algumas vantagens do MySql são: portabilidade (suporta as mesmas plataformas do Java), compatibilidade com as mais populares linguagens de programação, bom desempenho, facilidade de uso, além de oferecer recursos para controle de transações.[9]

Para realizar a conexão de SGBDs com o Java, ele utiliza o driver de conexão JDBC. Existe um conector de JDBC para MySql, que consiste em um arquivo Jar que deve ser adicionado às bibliotecas do projeto Java.[9]

MySql foi o SGBD escolhido na implementação da PGCP pelo fato de oferecer integração com Java a, pela facilidade de uso, por ser um SGBD de código livre, além de ser uma ferramenta robusta e vastamente utilizada.

2.3.4 Protocolos de Rede

Na camada de transporte do modelo OSI, estão localizados os protocolos TCP e UDP, os quais são de grande importância para os diferentes tipos de fluxo de dados da PGCP.[10]

O protocolo TCP tem como principal objetivo a confiabilidade na entrega dos pacotes. A comunicação é realizada criando-se uma espécie de conexão virtual entre dois computadores. Nesta comunicação, os dois computadores podem tanto enviar quanto receber pacotes um do outro, caracterizando o que chamamos de conexão *full-duplex*. Segundo Daniel G. Costa [7]: "Antes de enviar qualquer dado utilizando o protocolo TCP, é obrigatório que uma conexão seja criada. Para isso, é necessária a troca de três segmentos: o primeiro com o bit SYN (sincronização inicial da conexão) marcado, o segundo com os bits SYN e ACK (segmento de aceitação, reconhecimento) marcados, e o terceiro segmento apenas com o bit ACK marcado. O primeiro e terceiro segmentos são originados do host cliente, enquanto que o segundo segmento é enviado pelo host servidor. Esse processo de abertura de conexão é conhecido como '*three way handshake*"'. Após a tranferência dos dados, a conexão é fechada com cada um dos hosts enviando um bit FIN (encerramento da conexão) seguido por um ACK do outro host, indicando que o fechamento da conexão foi recebida.[7]

O protocolo UDP, ao contrário do TCP, não utiliza o processo de *three way handshake*. Na conexão UDP, não há qualquer garantia de que o pacote foi entregue. Um host apenas envia na rede os dados com o IP e a porta de destino. Se a porta neste IP estiver aberta e aguardando mensagens UDP, o datagrama (unidades de transmissão do UDP) é recebido, caso contrário ele é perdido. Em uma pegunta sobre qual o melhor protocolo, instintivamente a resposta seria o TCP, já que este oferece maior segurança de que a informação irá chegar ao seu destino. Porém a resposta estaria errada, pois a resposta certa para esta pergunta é "depende". Para uma transferência de um arquivo de um servidor FTP (utiliza TCP) por exemplo, o protocolo TCP é realmente o mais indicado, pois garante a entrega de cada pacote contido no arquivo. Todavia, se a aplicação usada depende de uma resposta em tempo-real, o TCP provavelmente iria apresentar muito atraso, praticamente inviabilizando o uso da aplicação. Entretanto com o protocolo UDP, tais aplicações se tornam viáveis, pois como o UDP é relativamente simples se comparado ao TCP, não oferecendo qualquer garantia de que o datagrama será entregue, ele obtém um grande ganho de velocidade na conexão.[7]

Na PGCP, ambos protocolos foram utilizados. O TCP foi utilizado na implementação de todas as comunicações entre clientes e o servidor. O protocolo UDP foi utilizado na áudio e videoconferência (cliente a cliente), devido a necessidade de estes recursos serem executados em tempo real. O protocolo UDP na verdade é utilizado como base para outro protocolo, o RTP (*Real Time Protocol*, protocolo de tempo real). O RTP usa o UDP como protocolo de transporte.

2.4 Plataformas Existentes

Existem no mercado inúmeras plataformas para gerência de projetos. Nesta seção serão apresentadas algumas das mais usadas. Duas delas são de código livre, e as outras duas de código proprietário.

2.4.1 Planner

O Planner é uma plataforma de código livre. Ele possui versões para Windows e para Linux, porém foi desenvolvido pela comunidade GNOME do Linux. É capaz de gerenciar projetos exibindo fases no gráfico de Gantt e gerenciando recursos ligados ao projeto. [11]

Esta ferramenta possui a vantagem de ser de código livre, além de ser multiplataforma. Por outro lado, ela não conta com ferramentas colaborativas e não possibilita a geração

de relatórios.

Na Figura 4 é ilustrada a interface do Planner[11].

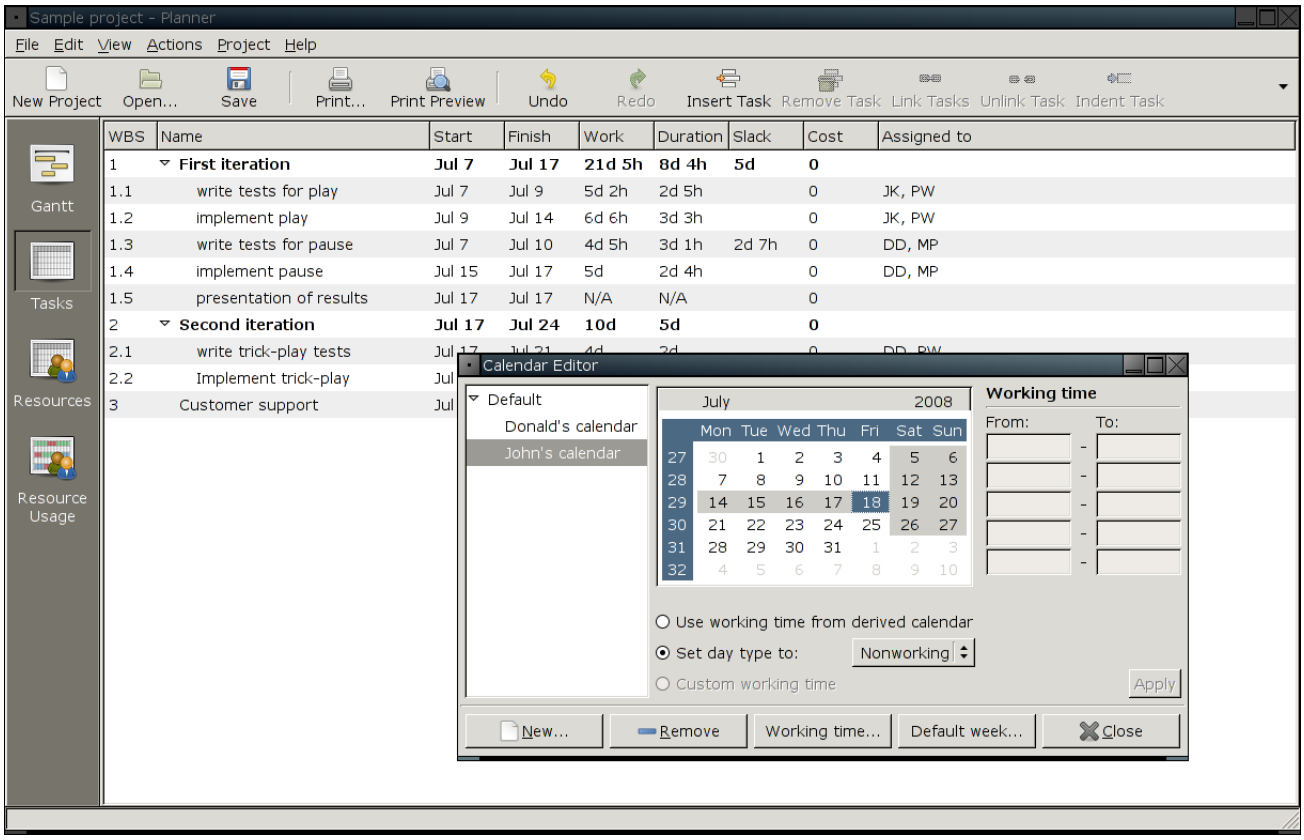

Figura 4: Interface - Planner

2.4.2 OpenProj

O OpenProj é um software para gerenciamento de projetos de código livre. Ele é capaz de abrir projetos do Microsoft Project. Possui gráfico de Gantt, gráfico PERT (um tipo de gráfico usado para planejamento), relatórios de uso das tarefas, custo do valor agregado, além de contar com gráficos de Estrutua Analítica de Recuros (EAR) e de Estrutura Analítica de Projeto (EAP). Ele não possui recursos colaborativos. [12]

O OpenProj se destaca pois, além de ser uma ferramenta de código livre, é capaz de abrir projetos do Microsoft Project e gerar alguns relatórios básicos. Entretanto, ela não possui ferramentas colaborativas.

A interface do OpenProj é apresentada na Figura 5[12].

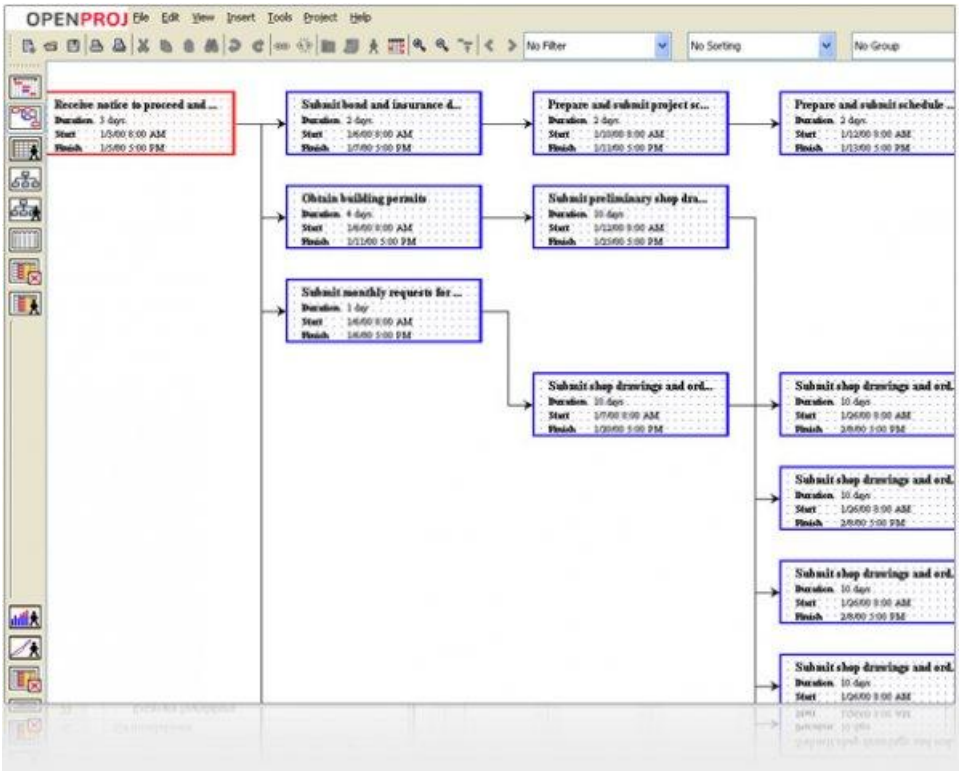

Figura 5: Interface - OpenProj

2.4.3 Microsoft Project

O MS Project é capaz de controlar as tarefas, agendamentos e finanças de projetos. Possui assistentes de modelos, opções avançadas de relatórios, além de poder ser conectado ao Microsoft Office Project Server, o que habilita os recursos de gerenciamento de projetos corporativos. Ele está disponível em sua versão 2010 a partir de U$ 999,95 . [13]

Apesar do MS Project ser um software proprietário e não multiplataforma, ele possui uma grande popularidade, muito devido ao fato da fácil utilização seguindo o ambiente Office. Ela também é capaz de gerar relatórios e possui algumas ferramentas colaborativas.

Na Figura 6 é ilustrada a interface do Microsoft Project[13].

2.4.4 Basecamp

O BaseCamp é uma ferramenta online de gerenciamento colaborativo de projetos. Possui ferramentas colaborativas como compartilhamento de arquivos, chat, notificações via e-mail, comentários e agenda. Possui suporte para 14 idiomas e possibilita o gerenciamento de projetos até mesmo através de celulares. O pacote mais barato custa 49 dólares por mês e permite a criação de 35 projetos, oferece 15 GB de storage e usuários ilimitados.

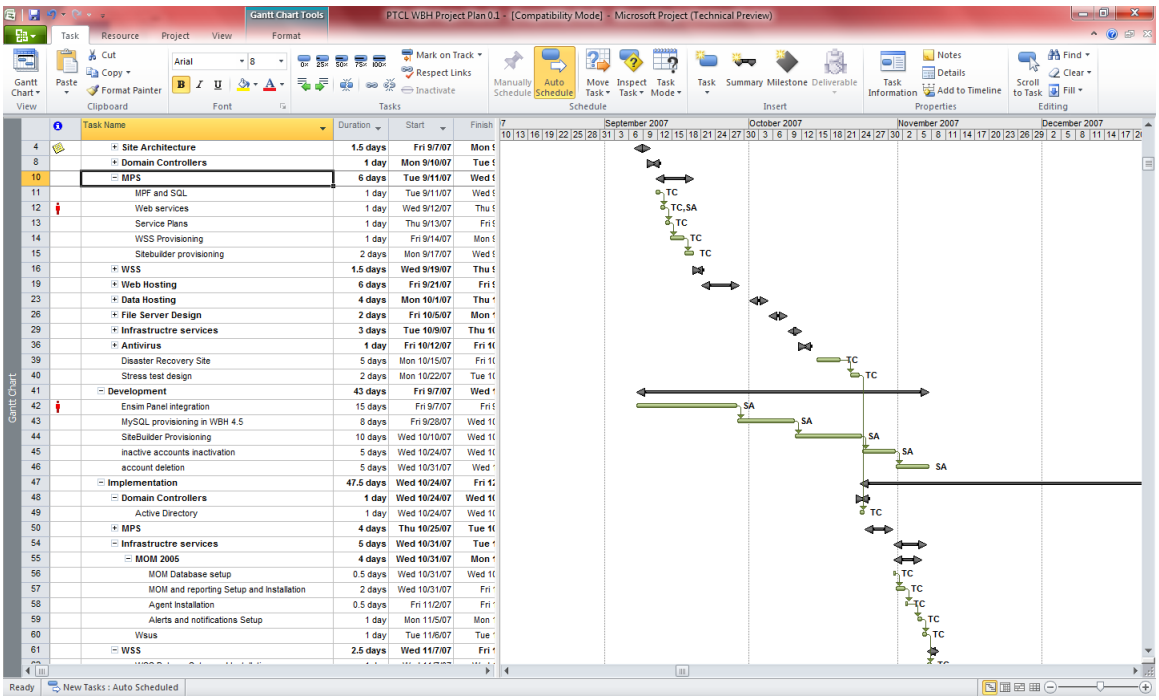

Figura 6: Interface - Microsoft Project

É possível fazer um teste gratuito de 30 dias sobre qualquer versão do sistema.[14]

O BaseCamp é uma ferramenta multiplataforma que contém algumas ferramentas colaborativas embutidas. No entanto, ele é um sistema proprietário e totalmente dependente da internet.

Um exemplo da interface do BaseCamp pode ser visto na Figura 7[14].

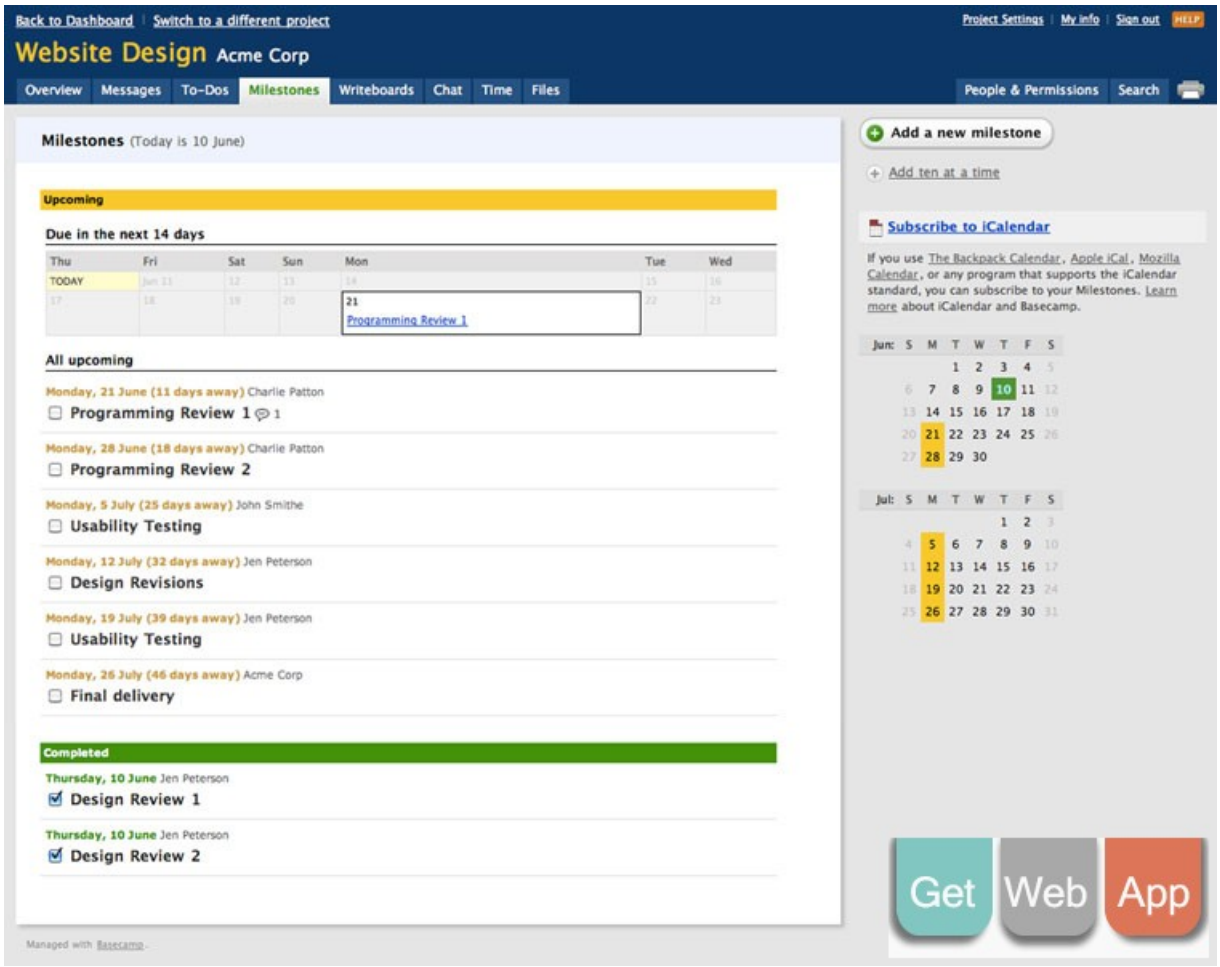

Figura 7: Interface - BaseCamp

Capítulo 3

PGCP - Plataforma para Gerenciamento Colaborativo de Projetos

Neste capítulo, são apresentadas as características, ferramentas e artefatos do projeto de software da PGCP, além de conter uma explicação detalhada de como a estrutura do sistema funciona.

O foco desta plataforma é a gerência de projetos de maneira colaborativa, ou seja, com auxílio de ferramentas interativas e colaborativas, visando maior produtividade e independência da posição geográfica do usuário.

A PGCP permite que usuários possam criar e administrar diversos projetos, onde cada projeto possui uma listagem de todos os usuários envolvidos, além de uma interface baseada em fases, onde cada fase registrada no sistema representa uma fase de um determinado projeto. Fases também podem conter atividades, que podem ser tratadas como sub-fases. Usuários podem utilizar as ferramentas colaborativas tanto sobre fases como sobre atividades, exceto repositório de documentos, o qual pode ser utilizado apenas nas fases. Para realizar a manipulação de projetos, existem usuários com diferentes privilégios, os quais serão abordados no início da seção 3.1. Usuários podem se comunicar de diversas formas, descritas da seguinte forma:

- Mensagens: usuários podem trocar mensagens internas entre si independente de estarem no mesmo projeto ou não. Esta troca de mensagem é realizada em um modelo semelhante a um *e-mail*, onde cada usuário contem uma caixa de entrada com mensagens.

- Bate-papo: quando, por exemplo, dois usuários estão em um mesmo projeto, estes

podem se comunicar através de um bate-papo, que funciona de maneira semelhante às salas de bate-papo existentes na internet. A restrição para isto é que, se tais usuários não forem administradores, eles só poderão entrar em um mesmo bate-papo se ambos estiverem atribuídos à mesma fase ou atividade.

- Comentários: cada fase ou atividade pode receber comentários de usuários que estão responsáveis por aquela fase ou atividade.

- Áudio e videoconferência: usuários podem realizar reuniões virtuais com a utilização da áudio e videoconferência. Tais recursos podem ser utilizados tanto em fases quanto em atividades, e também são restritos a usuários que são responsáveis pela fase ou atividade correspondente.

- Repositório de Documentos: pode haver um repositório para cada fase, onde usuários podem realizar *upload* (enviar) ou *download* (baixar) de arquivos.

Para que a PGCP troque informações pela rede com outros usuários, é necessária a implementação de dois módulos: módulo servidor e módulo cliente. O módulo servidor deve ser aberto em uma instância única, o qual é responsável por manter uma lista de clientes *online*, autenticar os clientes, intermediar alguma operações entre eles, além de esperar a conexão de novos clientes. No módulo cliente, o usuário uma vez autenticado pelo servidor, acessa a base de dados diretamente, sem intermédio do servidor. A partir daí o cliente obtém acesso ao sistema que o permite realizar as tarefas necessárias. Não é possível executar o módulo cliente sem que haja um módulo servidor sendo executado, localmente ou em outra máquina. A base de dados pode estar localizada em qualquer computador, não necessariamente no mesmo que o módulo servidor esteja sendo executado, apenas sendo necessário que o cliente e o servidor consigam roteá-lo, ou seja, assumindo que o módulo cliente, o módulo servidor e a base de dados estão em computadores distintos, é necessário que os três possuam IPs reais válidos ou que estejam dentro de uma mesma "rede falsa"(NAT), onde cada um dos três computadores possam "enxergam"uns aos outros.

3.1 Desenvolvimento do sistema

Nesta seção são apresentados detalhes da implementação da PGCP, explicando de que forma são realizadas requisições do cliente ao servidor, como estes dados são tratados no servidor e como são tratadas as respostas do servidor no cliente. No decorrer da seção,

alguns artefatos considerados relevantes e que auxiliam no entendimento são apresentados. São eles: diagrama de casos de uso, diagramas de sequência, diagramas de colaboração, diagramas de classes e modelagem das tabelas do banco de dados.

A PGCP é dividida em dois módulos: cliente e servidor. No cliente, existem dois tipos de usuário de *sistema*, o administrador e o usuário comum. O administrador possui privilégios para fazer toda e qualquer tarefa no sistema, desde a criação de um novo usuário até a manipulação de projetos. Já o usuário é limitado a enviar mensagens para outros usuários, podendo também visualizar projetos apenas nos quais ele está atribuído e utilizar as ferramentas colaborativas de fases e atividades de projetos apenas das quais ele participa.

Existe também o tipo de usuário administrador de *projeto*. O administrador de projeto tem total acesso apenas ao projeto ao qual ele administra, e não a todos. Para um usuário comum se tranformar em administrador de um determinado projeto, um administrador do *sistema* ou um outro administrador do *projeto* deve assim nomeá-lo. A partir do momento da nomeação, o usuário, que a nível de *sistema* é um usuário comum, passa a ter privilégios de administrador dentro de um *projeto*. Esta funcionalidade ajuda na divisão de tarefas entre gerentes de projeto, podendo até separar projetos muito grandes em projetos menores e atribuir cada projeto menor a um membro da equipe.

Tanto no módulo servidor como no módulo cliente, toda e qualquer requisição à base de dados é realizada através de classes que se comunicam exclusivamente à base de dados. Na realização de uma requisição, primeiramente é criada uma instância da classe *Conexão*, a qual contém as informações do banco de dados e a localização do driver de conexão. Posteriormente à conexão, é realizada a requisição a uma das classes de persistência. Tais classes contém métodos de inserção, atualização, deleção, dentre outros que foram fazendo-se necessários na implementação do sistema.

Na Figura 8 é ilustrado o diagrama de casos de uso, o qual contém os atores do sistema e as operações que cada um pode realizar. Os diagramas apresentados nesta monografia foram feitos utilizando a ferramenta para criação de UML chamada *Astah*.

Na Figura 8, pode-se observar a existência de setas entre os atores. Tal notação serve para simplificar a construção do diagrama de casos de uso, podendo isolar as funcionalidades por atores, sem ter que repetí-las quando um ator possuir as mesmas funcionalidades de outro.

A seta que sai do ator Usuário e vai até o Administrador projeto indica que todas as

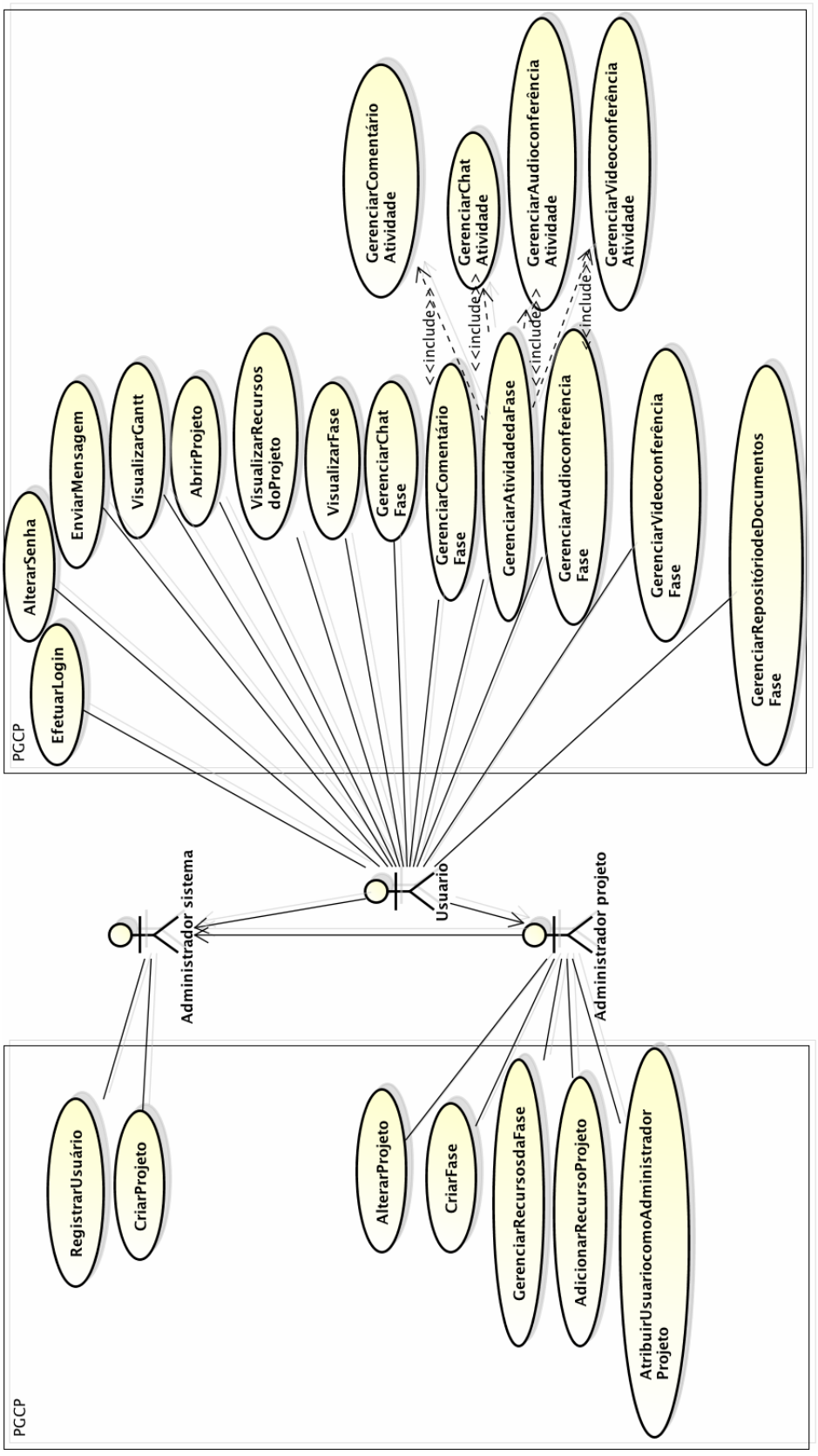

Figura 8: Diagrama de Casos de Uso.

operações que o Usuário tem permissão para realizar, o Administrador projeto também tem. A seta que vai do Administrador projeto ao Administrador sistema indica que o Administrador sistema possui permissão para realizar todas as operações do Administrador projeto, ou seja, o Administrador sistema é o usuário com maior nível de permissão no sistema.

Todos os casos de uso apresentandos no diagrama são correspondentes ao módulo cliente da PGCP. O módulo servidor não possui qualquer interação com o usuário, não possuindo atores e nem casos de uso.

A fim de organizar a explanação da PGCP, os diagramas de classes do módulo cliente e servidor, respectivamente representados nas Figuras 9 e 10, são divididos em pacotes, isolando alguns recursos colaborativos que requerem uma explicação mais detalhada. Esta seção é dividida em subseções, cada qual apresentando as características e explicações de cada pacote dos diagramas de classes.

A modelagem das tabelas do banco de dados é ilustrada pela Figura 11.

3.1.1 Estrutura

Todas as funcionalidades descritas a seguir tratam exclusivamente das classes contidas no pacote Estrutura dos diagramas de classes.

Para iniciar um cliente, primeiro é necessário executar uma instância do módulo servidor. No módulo servidor, a primeira classe invocada ao iniciar é a classe *Servidor*, a qual possui o método chamado *rodar* que inicializa uma instância da classe *ServerSocket*. A classe *ServerSocket* é a responsável por aceitar novos clientes, e recebe como parâmetro no construtor a porta e o número máximo de clientes. Posteriormente, uma *thread* chamada *EsperaConexoes* é criada. Essa *thread* é responsável por esperar conexões de novos clientes, validá-los e conectá-los.

3.1.1.1 Autenticação

Quando uma instância do módulo cliente é aberta, uma interface de *login* é apresentada, a qual solicita ao usuário os atributos usuário e senha. O atributo abaixo da senha, o qual pede para que o usuário selecione o adaptador de rede que faz a conexão com a internet, é utilizado nas ferramentas de áudio e videoconferência. Este atributo é necessário pois o usuário pode possuir diversas interfaces de rede em seu computador, cada uma com um IP, onde pode acontecer de uma das placas não realizar conexão com a internet.

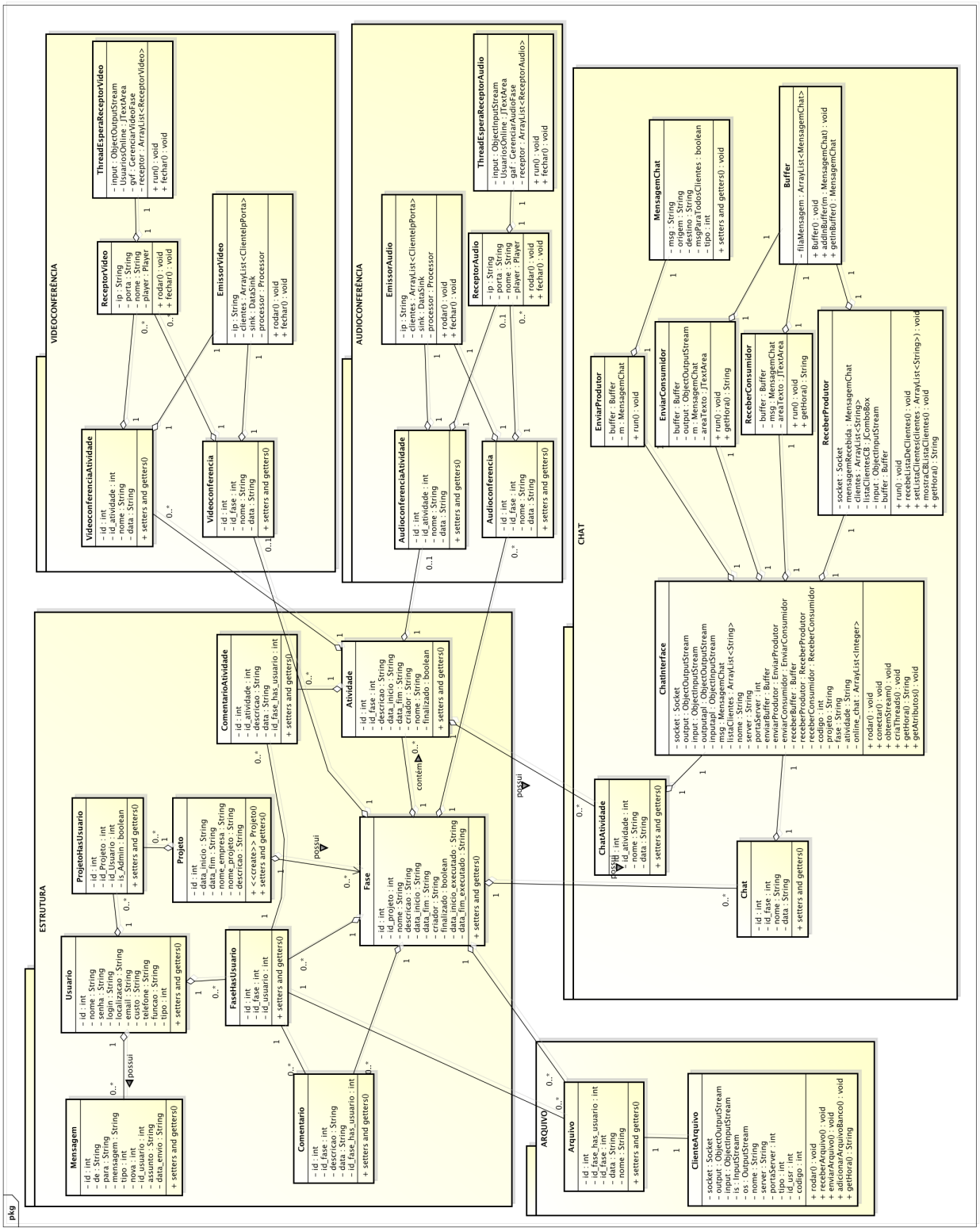

Figura 9: Diagrama de Classes - PGCP Cliente

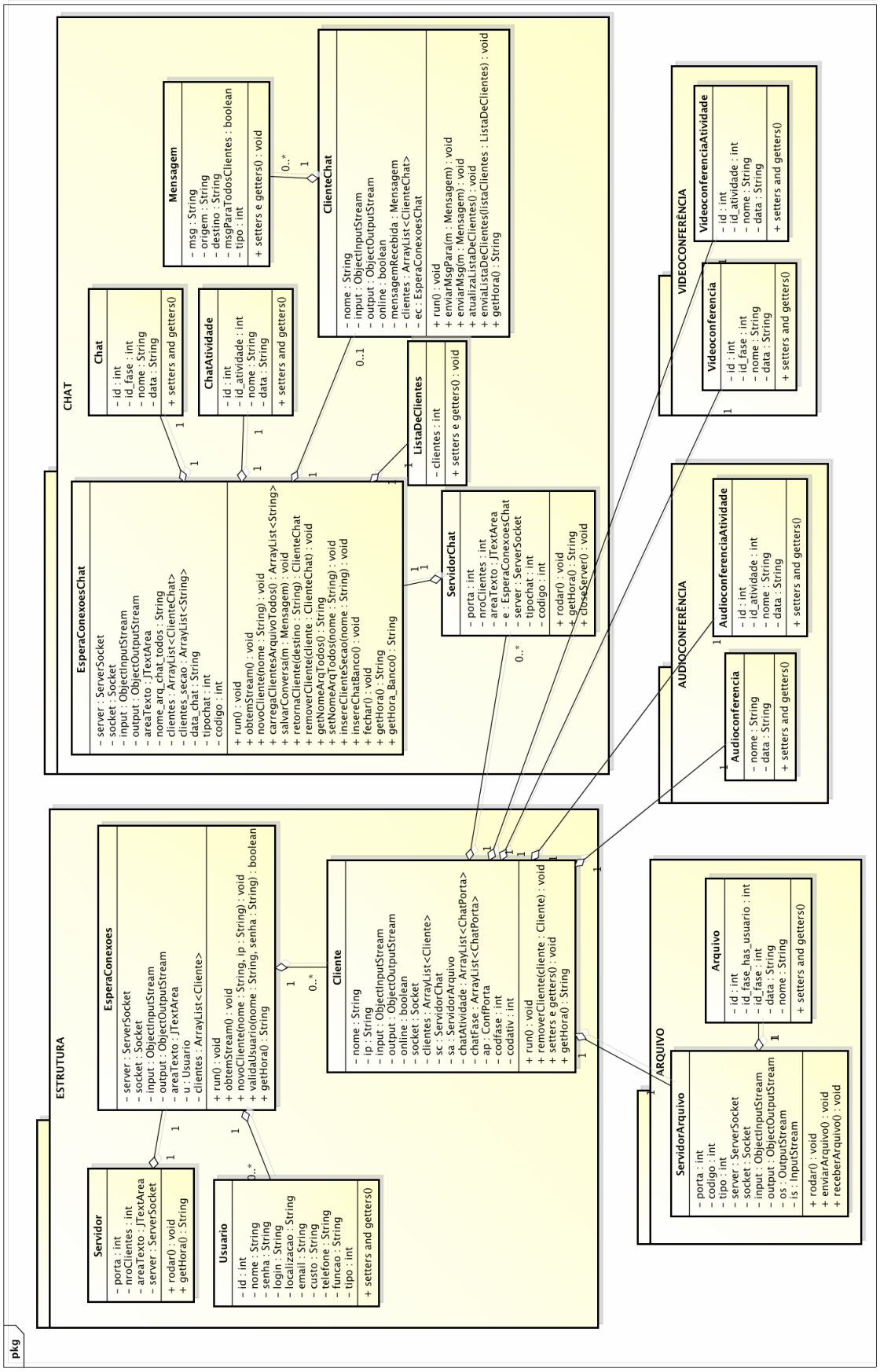

Figura 10: Diagrama de Classes - PGCP Servidor

Figura 11: Modelagem da Base de Dados

Desta maneira, é requerido ao usuário que aponte qual a interface de rede, seguida do IP atribuído a ela no momento, que realiza a conexão com a internet. As ferramentas de áudio e videoconferência serão aboradas com mais detalhes em 3.1.4, porém justificando a necessidade deste atributo de IP, quando dois clientes criam uma conferência, o módulo servidor apenas armazena em qual IP e porta há transmissões naquela determinada fase ou atividade. A função do servidor é apenas armazenar tais dados e repassá-los para os demais participantes (clientes). Já a conexão para a realização da conferência é feita entre os clientes diretamente. Para isto, é necessário apenas que o servidor tenha armazenado os IPs dos *hosts* que realizam a conexão com a internet.

Após a entrada do usuário, senha e IP, o módulo cliente inicia uma conexão entre ele e o módulo servidor, especificando o IP e a porta do servidor para isto. Esta solicitação de conexão é realizada através do uso da classe *Socket*, a qual é a responsável pela troca de mensagens entre cliente e servidor, numa conexão do tipo *full-duplex*, ou seja, cada lado da conexão pode receber e enviar mensagens ao outro lado. Quando a classe *Socket* é instanciada no lado do cliente, a classe *EsperaConexoes* do servidor, a qual espera novas conexões, é acionada, aceitando a conexão. Até agora, apenas o canal de comunicação entre cliente e servidor foi criado. O envio e recebimento de mensagens entre cliente e servidor é realizado por dois atributos da classe *Socket*, o *input* e o *output*. O *input* é responsável por receber informações, ao passo que o *output* tem a função de enviar.

Após a aceitação da conexão do cliente por parte do servidor, o cliente envia, através do *output*, o nome de usuário e a senha ao servidor. No servidor, após o recebimento destes atributos através do *input*, o método *validaUsuario* da classe *EsperaConexoes* acessa a base de dados e verifica se o nome de usuário e senha recebidos conferem. Se não for autenticado, o servidor envia uma mensagem ao cliente avisando sobre o erro. Se autenticado, o servidor envia ao cliente o objeto do tipo *Usuario*, o qual contém todas as informações pessoais do usuário. Uma vez autenticado, o cliente envia o IP ao servidor para que este o armazene também. O servidor, por sua vez, usa as informações de nome e IP recebidas e invoca o método *novoCliente*, o qual inicia uma *thread* chamada *Cliente*. É nesta *thread* que todas as demais requisições ao servidor serão tratadas. No lado do cliente, após o envio do IP ao servidor, a autenticação é finalizada e a interface principal do sistema é apresentada.

O Diagrama de sequência contido na Figura 12 é ilustada a operação de efetuar login.

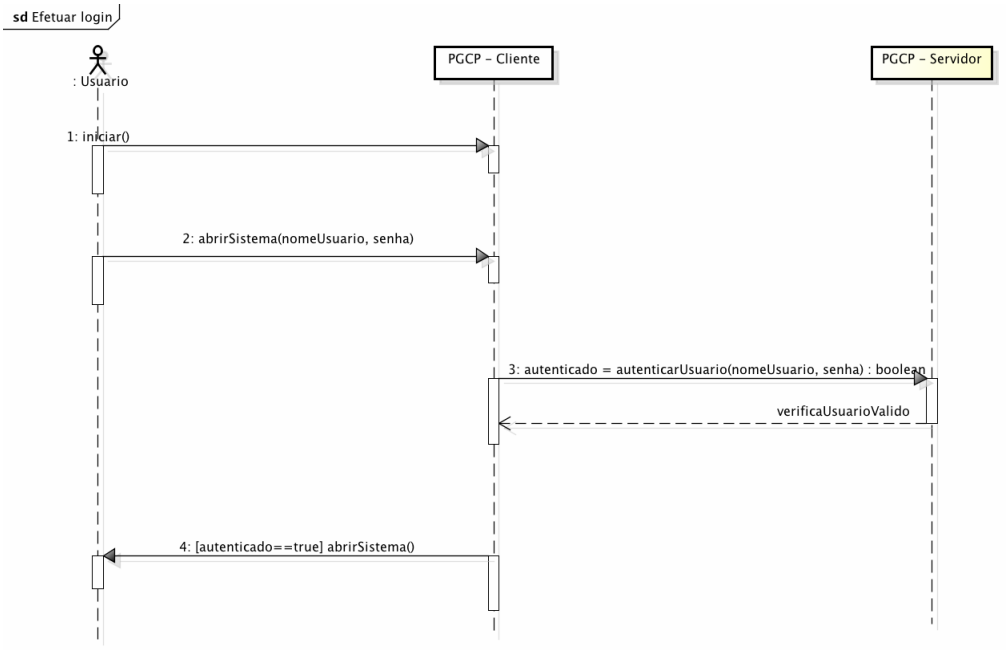

Figura 12: DSS Efetuar Login

3.1.1.2 Gerenciar Usuários

A PGCP pode ser dividida em 3 áreas: gerência de projeto (a mais ampla), correio eletrônico e gerência de usuários.

Como visto no início deste capítulo, após a autenticação do usuário, o módulo cliente passa a acessar a base de dados diretamente, sem intermédio do sevidor.

Todos os tipos de usuário tem acesso à área de gerência de usuários, porém usuários comuns podem apenas visualizar informações dos outros usuários e alterar a própria senha, ao passo que administradores do sistema tem permissão total. A operação de alterar a senha, a qual é ilustrada pelo caso de uso AlterarSenha, é realizada de forma bastante simples. O *Frame GerenciarUsuarios* possui um *Frame* interno chamado *AlterarSenha*. Neste *Frame* é requisitado ao usuário a inserção da senha atual e de uma nova senha. Após a confirmação do usuário, a aplicação captura a nova senha e a passa para a classe PersistenciaUsuario, a qual, através do método *update*, realiza a atualização da tabela da base de dados.

A operação de registrar um novo usuário pode ser realizada apenas pelo administrador do sistema e está contida na classe *NovoUsuário*. Esta classe é uma extensão de um *Frame* que contém atributos que devem ser inseridos com as informações do novo usuário. Os atributos a serem informados são: nome, login, senha, localização, telefone, e-mail, função, tipo (administrador ou usuário comum) e custo. Após o preenchimento,

um objeto da classe *Usuario* é montado e então é invocado o método *insert* da classe *PersistenciaUsuario*, passando o objeto da classe *Usuario* como parâmetro. O método *insert* então realiza a inserção do novo usuário na base de dados.

3.1.1.3 Correio Eletrônico

A área de correio eletrônico funciona como um sistema de e-mail interno do sistema. Nela é possível realizar troca de mensagens entre usuários do sistema, sem que seja necessário que os usuários destinatários estejam *online*. As mensagens de cada usuário são mantidas em uma tabela da base de dados. A aplicação dá opções de manejamento das mensagens, como a possibilidade de enviar mensagens a mais de um destinatário de uma só vez, responder uma mensagem, encaminhar, ou responder a todos. Todas estas opções funcionam de maneira semelhante, se diferenciando apenas por alguns detalhes.

A operação de enviar uma nova mensagem encontra-se na classe *MeuCorreio*, a qual invoca a classe *EnviarMensagem*. Esta classe é uma extensão de um *Frame* que contém atributos como destinatários, assunto, tipo da mensagem (importante ou não) e a mensagem em si. Para enviar uma mensagem a mais de um usuário, basta inserir o nome dos usuários no atributo Para, separando-os por vírgula. Porém usualmente não se sabe o nome de usuário de todos. Para resolver este problema, é possível selecionar a opção Para, a qual leva a um *Frame* que realiza uma consulta à base de dados e lista todos os usuários do sistema juntamente com o nome de cada um. Desta forma, o usuário pode adicionar todos destinatários, um a um. A cada destinatário adicionado, a aplicação monta uma *String,* separando cada usuário por vírgula. Ao finalizar a seleção dos destinatários, a *String* com os destinatários é copiada para o atributo Para, e quando finalmente a mensagem é enviada, a aplicação captura todos os atributos inseridos, separa cada nome do atributo Para em uma *String* diferente, introduzindo-a em um *ArrayList*. Por fim, todos os nomes deste *ArrayList* são percorridos um a um, realizando uma requisição ao banco de dados para capturar o código de cada destinatário, e desta forma montando um objeto da classe *Mensagem* e passando-o para a classe *PersistenciaMensagem*, a qual realiza a inserção da mensagem no correio de cada destinatário.

A ação de enviar mensagem pode ser ilustrada pelo diagrama de sequência contido na Figura 13 e diagrama de colaboração da Figura 14:

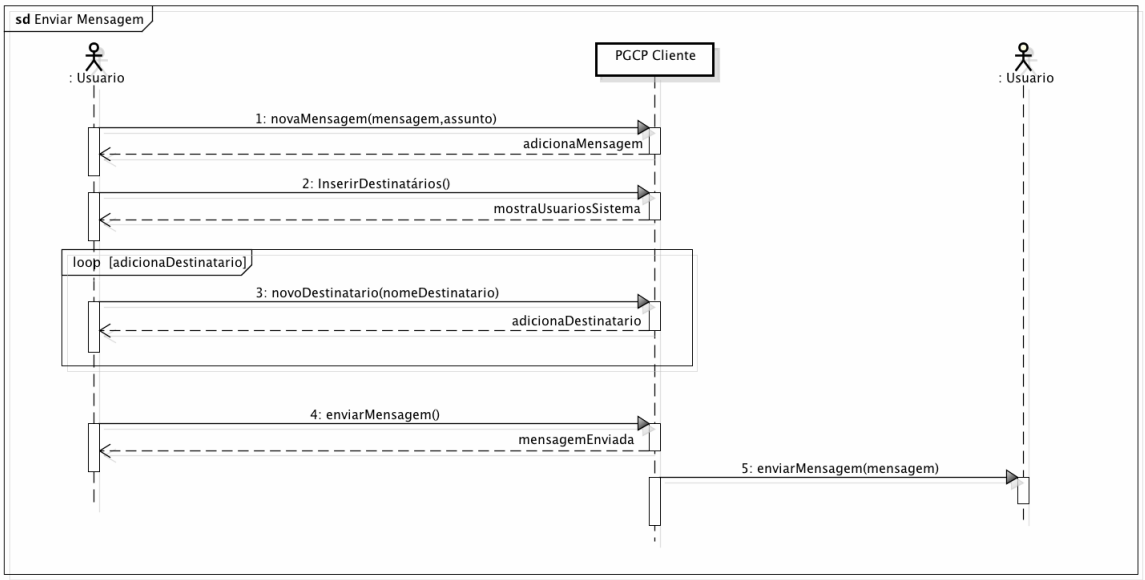

Figura 13: DSS Enviar Mensagem

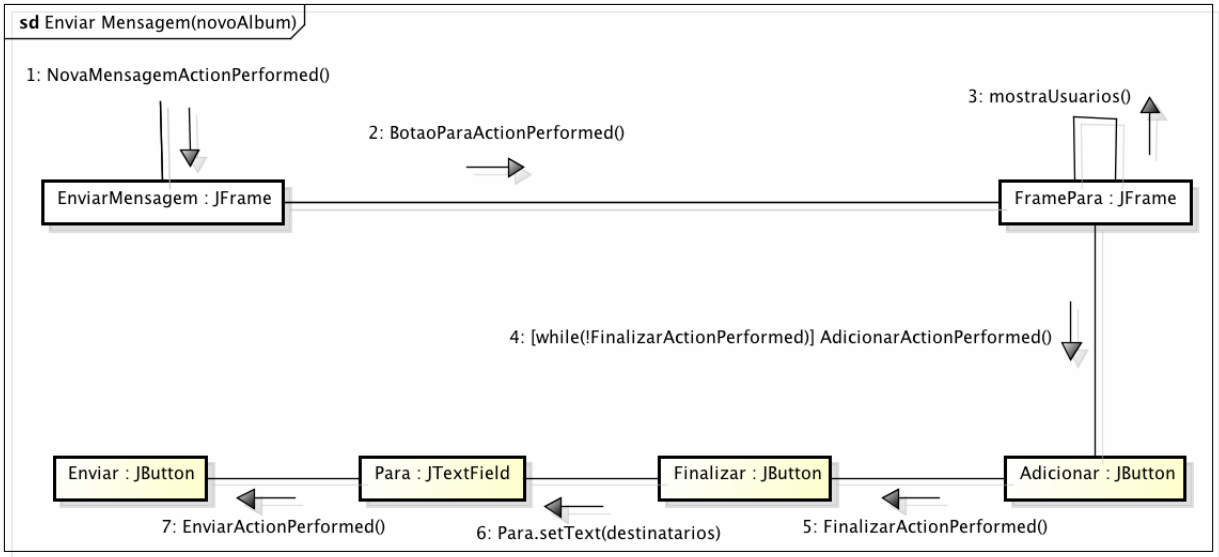

Figura 14: Diagrama de Colaboração - Enviar Mensagem

3.1.1.4 Operações sobre Projeto

Todas as operações restantes se referem à gerência de projeto. As operações mais básicas, como criação, abrimento e alteração de projetos, são descritas inicialmente.

A operação de criação de um projeto é exclusiva dos administradores do sistema. Após o usuário se autenticar, é realizada uma verificação para determinar se ele é administrador do sistema ou não. Esta verificação é feita através do atributo *Tipo* do usuário. Se o usuário for administrador do sistema, a opção de novo projeto é habilitada. Quando acionada, esta opção leva o usuário a um *Frame* onde são requisitadas informações como nome do projeto, nome da empresa e descrição do projeto. Finalizado o preenchimento dos atributos, a aplicação captura-os, realiza a instanciação de um objeto da classe *Projeto*, e o envia para a classe *PersistenciaProjeto*, a qual fica responsável por inserir o projeto na base de dados. Após a inserção, a interface de gerenciamento de projeto (classe *TelaProjeto*) é apresentada.

A alteração de um projeto é realizada por administradores de projeto e sistema apenas. Se o usuário logado for um administrador do sistema, a alteração é permitida. Porém se não for, é necessário verificar se o projeto selecionado tem como um dos seus administradores o usuário logado. Está verificação é realiza capturando-se o código do projeto selecionado e o código do usuário logado. Estes códigos são utilizados para montar um objeto da classe *ProjetoHasUsuario* e enviar ao método *verificaAdmin* da classe *PersistenciaProjetoHasUsuario*. Este método acessa a base de dados, verificando se o usuário logado está no projeto selecionado. Se estiver, o método retorna um objeto *ProjetoHasUsuario* que contém um atributo boleando *is_Admin*. Se o valor deste atributo for verdadeiro, a alteração é permitida, pois o usuário logado é um dos administradores do projeto selecionado. Se for falso ou a consulta no banco de dados verificar que o usuário não está no projeto, a alteração é negada. A alteração é feita de forma semelhante a criação de um projeto, com a diferença de que na alteração é possível especificar a data de início e término do projeto, e por fim realizando um update na tabela referente ao projeto alterado.

Na abertura de um projeto, uma única verificação é realizada afim de conhecer o tipo de usuário logado (administrador do sistema ou usuário). Se for administrador do sistema, todos os projetos existentes são apresentados. Se for um usuário (aqui não há distinção entre administrador de projeto e usuário comum), é realizada uma consulta na base de dados invocando o método *getAllFromUser* da classe *PersistenciaProjetoHasUsuario*, o qual retorna todos os projetos nos quais o usuário logado está envolvido. Desta forma,

apenas estes projetos são apresentados ao usuário.

O processo de abertura de um projeto é ilustrado pela Figura 15 que contém o diagrama de sequência.

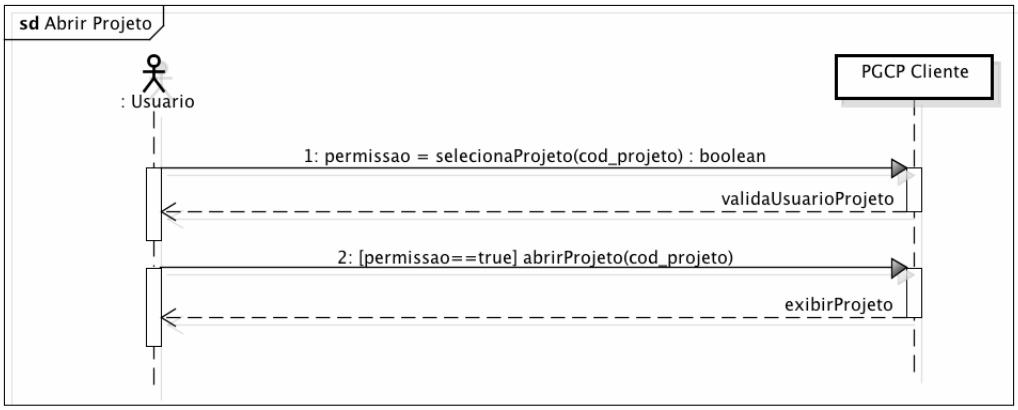

Figura 15: DSS Abrir Projeto

A exclusão de um projeto só pode ser realizada por administradores de sistema e de projeto. Para evitar deixar "lixo" na base de dados, é realizada uma busca no projeto, procurando por fases, atividades, comentários, audio e videoconferência e arquivos acoplados ao projeto, excluindo todos estes registros antes de realizar a deleção do projeto. Esta busca é realizada utilizando métodos de cada classe de Persistencia dos recursos mencionados há pouco. Estes métodos retornam *ArrayLists* contendo os códigos de cada recurso acoplado ao projeto a ser apagado.

Para evitar a repetição, o processo que acontece a nível de código em um simples preenchimento de formulários não será sempre descrito, apenas chamando atenção para particularidades da operação que será realizada.

A gerência de projeto na PGCP é realizada à base de criação de fases, as quais podem possuir atividades. Dentro destas fases e atividades, são utilizados os recursos colaborativos. Porém, para um projeto permitir que usuários utilizem estes recursos, estes usuários devem ser de alguma forma inseridos nos projetos, ou seja, deve-se determinar quais usuários do sistema participaração de um determinado projeto.

A nível de projeto, passa-se a chamar os usuários de recursos, onde adicionar um recurso ao projeto é o mesmo que adicionar um usuário ao projeto. Tal operação só pode ser realizada por administradores (de projeto ou de sistema). Esta operação acontece de forma similar à adição de destinatários a uma mensagem, onde é apresentado um *Frame* contendo todos os usuários do sistema. A cada usuário adicionado como recurso, eles

são diretamente inseridos em um *ArrayList* de String até o usuário finalizar a adição. Neste momento, o *ArrayList* é percorrido e para cada recurso adicionado, é criado uma instância da classe *ProjetoHasUsuario*, na qual são inseridos o código do projeto e o código do usuário a ser adicionado. Por fim, cada instância desta classe é enviada à base de dados através do método *insert* da classe *PersistenciaProjetoHasUsuario*.

O processo de adicionar um recurso a um projeto é ilustrado pelo diagrama de sequência contido na Figura 16.

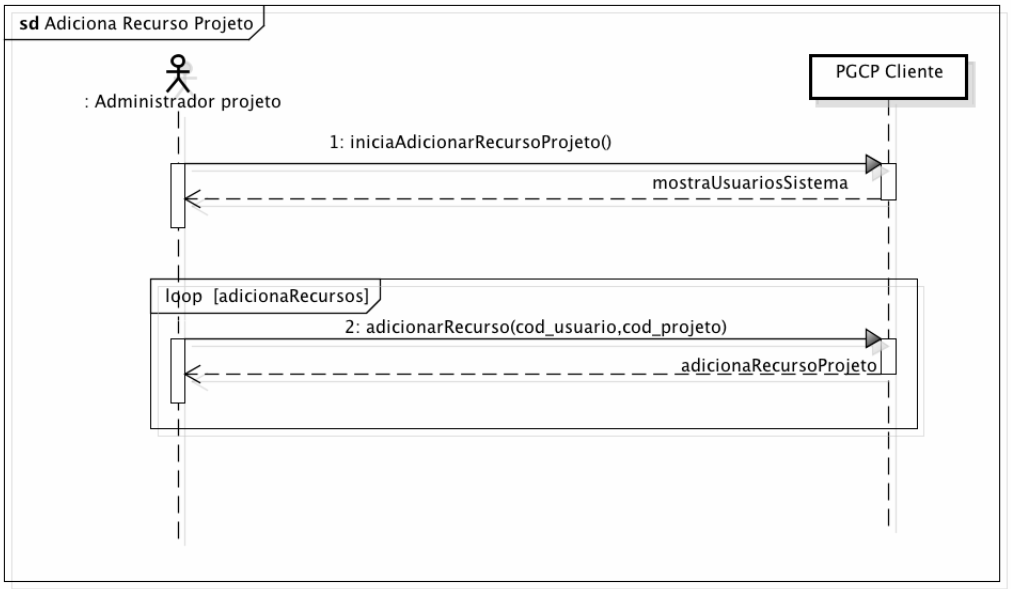

Figura 16: DSS Adicinar Recurso a Projeto

Um usuário comum, não administrador de um projeto, tem permissão apenas para visualizar os recursos do projeto no qual ele participa. A visualização é feita através de uma consulta à tabela da base de dados *ProjetoHasUsuário*, a qual retorna o código de todos os usuários inseridos em um projeto. Estes códigos então são utilizados para realizar uma consulta na tabela *Usuario*, retornando os dados de todos usuários, que são armazenados em um *ArrayList* contendo objetos da classe *Usuario*. Finalmente este *ArrayList* é percorrido e os recursos adicionados ao projeto são inseridos um a um em um objeto *JTable*.

Como visto na seção 3.1, existem usuários comuns que podem se tornar administradores de projetos se assim forem nomeados. Esta ação é realizada através da utilização da classe e também tabela do banco de dados chamada *ProjetoHasUsuario*. Para cada usuário adicionado como recurso em um projeto, um registro desta tabela é criado contendo o código do projeto e do usuário, além de um atributo boleano chamado *is_Admin* que define se o usuário é administrador do projeto ou não. Por padrão, um usuário adici-

onado como recurso de um projeto não é administrador dele. Portanto a ação de definir um usuário como administrador de um projetor limita-se a alterar este atributo para *true*. Isto é feito capturando-se o código do usuário selecionado para ser administrador e o código do projeto. Este par de códigos possibilita recuperar o registro na base de dados e alterar o atributo *is_Admin,* finalizando com um *update* no mesmo registro da base. Vale ressaltar que esta ação é reversível, ou seja, é possível retirar o privilégio de administrador de projeto de um usuário.

O processo de atribuição de um usuário como administrador de um projeto é ilustrado pelo diagrama de sequência contido na Figura 17.

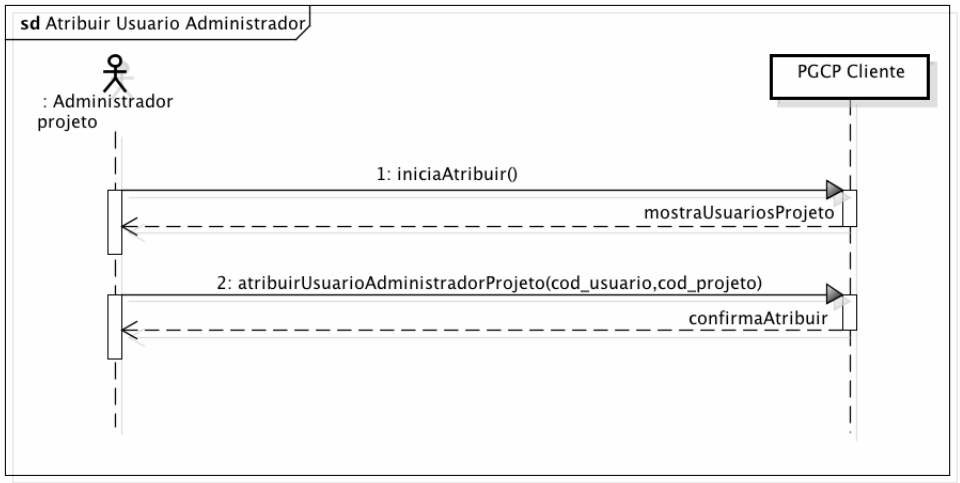

Figura 17: DSS Atribuir Usuário como Administrador de Projeto

A criação de uma fase só pode ser realizada por um administrador (de projeto ou sistema), o usuário não tendo permissão para tal ação. Nesta tarefa, a inserção dos dados é feita da mesma forma como nas outras operações. A particularidade da fase é que ela possui datas planejadas para sua execução e datas reais executadas, além de um indicador que mostra se a fase já foi finalizada ou não. Na criação da fase, apenas a data planejada é informada, as datas executadas ficam vazias (*null*). Na alteração das fases, as datas podem ser informadas e a fase finalizada, mas também podem não ser informadas quando a execução da fase ainda não foi iniciada.

Usuários comuns sem permissão de administrar um projeto podem apenas visualizar as fases, sem poder alterá-las. A visualização se dá pela operação de seleção de uma das fases, onde é realizada uma requisição dos dados da fase à base de dados.

A deleção de uma fase, assim como no projeto, requer uma pesquisa e deleção de todos registros que possuam qualquer relação com a fase a ser deletada.

A partir do momento em que há fases, é interessante delegá-las para recursos, afim de definir "quem fará o que". Na PGCP, recursos podem ser adicionados a diversas fases. Este ponto é importante, pois determina permissões de execução sobre as ferramentas colaborativas. Esta operação, a qual é ilustrada pelo caso de uso GerenciarRecursosdaFase, pode ser realizada apenas por administradores. Assim como na adição de um recurso a um projeto, onde é criado um registro na tabela da base de dados *ProjetoHasUsuario*, na atribuição de um recurso a uma fase, a tabela onde é criado um registro é a *FaseHasUsuario*, que indica o código da fase e o código do usuário.

O processo de adição de recursos em uma fase é ilustrado pelo diagrama de sequência contido na Figura 18.

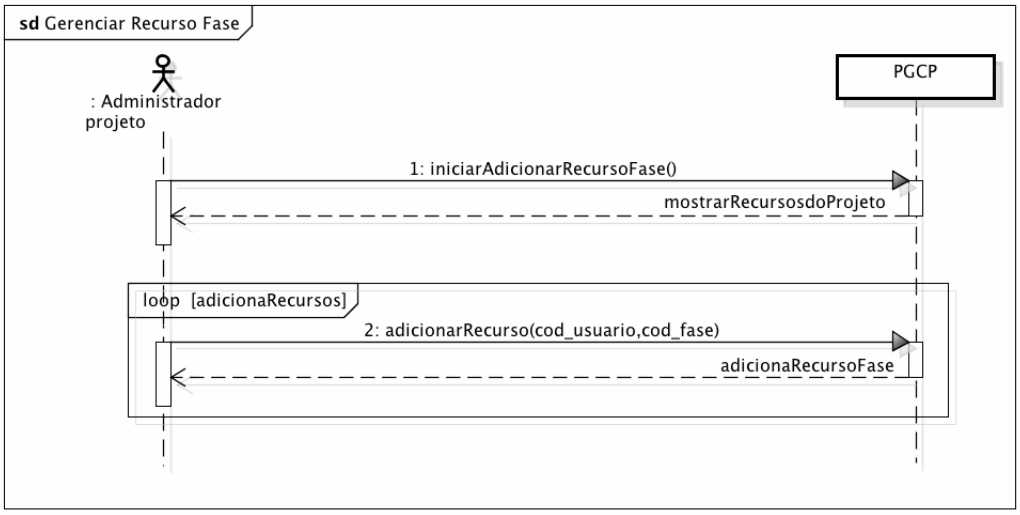

Figura 18: DSS Adicionar Recursos a uma Fase

Para realizar um acompanhamento visual das fases, a PGCP conta com um recurso bastante utilizado nos sistemas de gerenciamento de projetos, o gráfico de Gantt. Este é desenhado em barras horizontais sobre uma linha do tempo, onde cada barra representa o início e o fim de uma fase. Pode-se visualizar tanto o gráfico do planejamento das fases (utilizam as datas planejadas de cada fase) como o de execução (datas de execução) delas. Para desenhar o gráfico, é usada uma classe nativa do java chamada *Graphics*, a qual permite desenhar formas geométricas dentro de um *JPanel*. O desenho do gráfico através das datas planejadas e executadas é feito da mesma maneira, apenas utilizando parâmetros diferentes para o desenho, onde inicialmente é desenhada uma barra superior a qual caracteriza a linha do tempo. Para saber por qual data começar o desenho, é realizada uma consulta em todas as datas iniciais de todas as fases e selecionada a menor data entre elas. São desenhados então os meses dos anos, divididos em dias por porções menores de espaço. As datas, por fim, são desenhadas abaixo desta linha do tempo, uma após a outra.

As classes usadas para o desenho são as *PaneDesenhoPlanejado* e *PaneDesenhoExecutado*. Por se tratar de classes complexas e com muitas operações matemáticas com pixels, não é interessante discuti-las. Ao lado do gráfico, há uma tabela exibindo as fases do projeto, com as datas planejadas e executadas, para uma melhor visualização.

Cada fase da PGCP pode conter subfases, chamadas atividades. Elas ajudam no detalhamento de uma fase, permitindo dividi-la em porções menores. Exceto pelo repositório de documentos, todos os recursos colaborativos podem ser utilizados tanto nas fases como nas atividades. As atividades possuem atributos semelhantes aos da fase. Portanto, elas se tornam bastante semelhantes. A operação de gerenciar uma atividade contém outras operações, como adicionar atividade, alterar, excluir e visualizar, e também gerenciar e utilizar as ferramentas colaborativas. Usuários comuns podem gerenciar as atividades de fases apenas das quais ele participa. Todo o processo de criação, alteração, exclusão e visualização de atividades é similiar ao processos correspondentes na fase e seguem os mesmos padrões de permissões, portanto não serão detalhá-los.

O processo de adição de uma atividade em uma fase é ilustrado pelo diagrama de sequência contido na Figura 19.

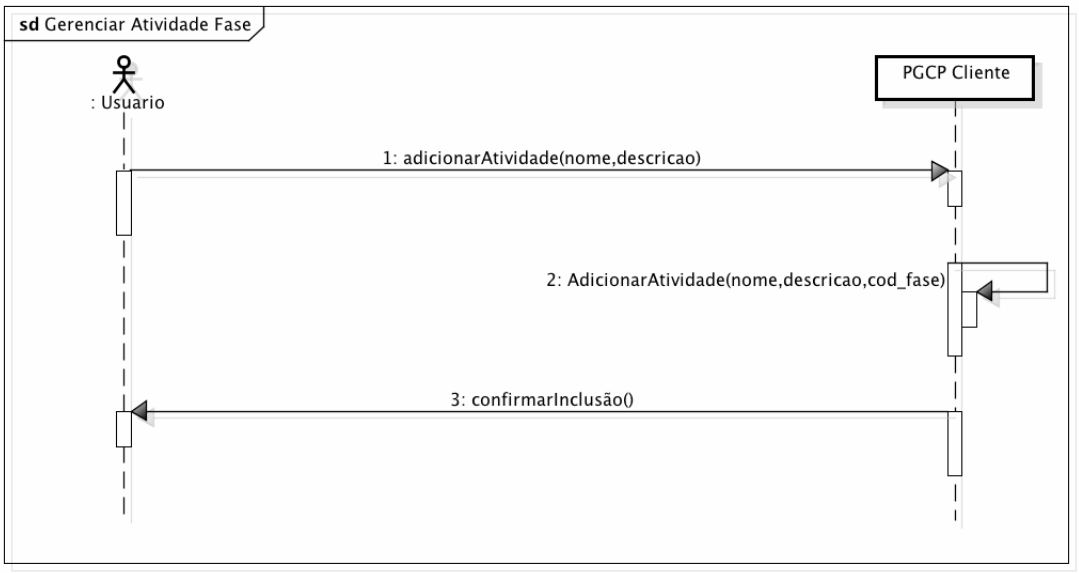

Figura 19: DSS Adicionar Atividade a uma Fase

No decorrer da utilização da PGCP sobre um projeto, usuários podem atualizar a situação de fases e atividades, escrevendo comentários sobre elas. Eles são utilizados de forma bastante simples, onde um usuário de uma fase ou atividade pode visualizar os comentários realizados pelos outros membros e também realizar seus próprios comentários.

O processo de adição de um comentário a uma fase é ilustrado pelo diagrama de

sequência contido na Figura 20.

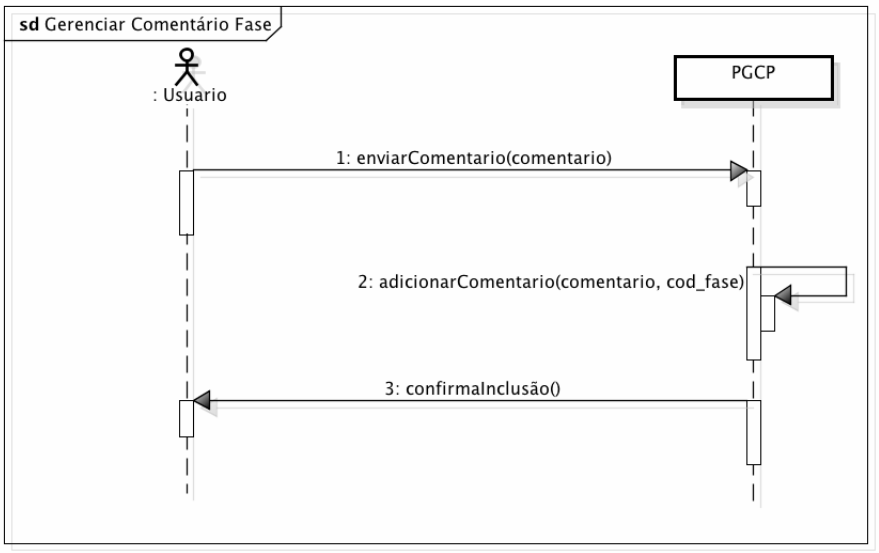

Figura 20: DSS Adicionar um Comentário a uma Fase

3.1.2 Bate-papo

Até o presente, o único momento no qual utilizou-se o *socket* para se comunicar com o servidor foi na autenticação. Isto ocorreu pois as operações realizadas até aqui não necessitavam trocar informações diretamente com outros clientes. As operações dos clientes eram armazenadas na base de dados e posteriormente apresentada a outros clientes. Porém, para implementar os recursos de *chat*, áudio e videoconferência e repositório de documentos, será necessário o uso de *socket* novamente, a fim de trocar mensagens entre servidor e clientes.

Como visto em 3.1.1.1, no servidor há uma *thread* chamada *Cliente*. Cada cliente conectado possui uma instância desta *thread* (classe). Toda *thread* possui um método chamado *run*, o qual fica em execução até que a *thread* seja fechada (método *stop*). Na classe Clientes, este método *run* fica em um *loop* infinito, sempre esperando por mensagens do módulo cliente da aplicação.

As requisições à classe *Cliente* são realizadas através de códigos, com cada código significando um recurso diferente. Quando o módulo cliente inicia um bate-papo, ele envia através do *output* o código 1, que quando é recebido pela *thread Cliente*, é identificado como uma requisição para iniciar um bate-papo. O segundo código que o módulo cliente envia é a indicação se o bate-papo a ser iniciado é um bate-papo em uma fase ou em uma atividade. O código 1 indica iniciar bate-papo em uma fase, 2 requisita desconectar

em uma fase, 3 requer o início de um bate-papo em uma atividade e 4 a desconexão na atividade.

Assumindo que o usuário deseje iniciar um bate-papo em uma fase, após enviar a requisição de início de um bate-papo em uma fase, o módulo cliente envia o código desta fase e fica aguardando o envio da porta a ser usada na conexão com o chat. No servidor, cada *thread Cliente* possui dois *ArrayLists* do tipo *ChatPorta*, uma classe simples que contém atributos código, porta e o objeto *ServidorChat* usado. Um *ArrayList* mantém os *chats* de fases e outro os de atividades. *ArrayLists* foram necessários pois um usuário pode participar de diversos bate-papos ao mesmo tempo. O servidor então inicia uma busca no *ArrayList* de bate-papos na fase de todos os clientes conectados (os quais são armazenados por um *ArrayList* de *thread Cliente*). Se o servidor encontrar um cliente que esteja com um bate-papo iniciado naquela fase, ele copia as informações do objeto *ChatPorta* do *ArrayList* de bate-papos de fase do cliente que já contém um bate-papo aberto naquela fase para o mesmo objeto do cliente que deseja se conectar. Por fim, o servidor envia a porta na qual está acontecendo o bate-papo para o módulo cliente. Se o servidor não encontrar qualquer usuário com um bate-papo iniciado naquela fase, ele terá que ser iniciado. Isto é feito através da obtenção de um porta livre através da classe *PortaLivre* (apenas retorna a porta seguinte de uma sequência), iniciando um *ServidorChat* naquela porta, montando o objeto *ChatPorta* e adicionando-o ao *ArrayList* de bate-papo fase e enviando a porta usada para o módulo cliente.

A partir do momento em que há um servidor de bate-papo sendo executado no módulo servidor e um cliente de bate-papo no módulo cliente, a conexão está pronta para ser iniciada. A comunicação entre o cliente e o servidor de bate-papo funciona de forma muito parecida com a comunicação do servidor e cliente da PGCP. A classe *ServidorChat* é responsável por criar o *ServerSocket* e iniciar uma *thread EsperaConexoesChat*. É importante ressaltar que para cada instância do bate-papo, é usado uma nova instância de *Socket*, para que o *Socket* da aplicação possa continuar realizando tarefas paralelamente. A *thread EsperaConexoesChat* tem a mesma função da classe *EsperaConexoes* do módulo servidor. Ela fica esperando requisições de conexões de novos clientes. No módulo cliente, após a porta ser recebida, é aberta uma instância da classe *ChatInterface*, onde inicialmente é aberto um *socket* com o *EsperaConexoesChat* na porta recebida e enviado o nome do usuário logado. Neste momento, a conexão do novo *socket* fica estabelecida e a troca de mensagens pode começar. Para simplificar a explicação, primeiro será explicado como funciona a parte cliente do bate-pao e posteriormente a parte servidor dele.

No envio de uma mensagem de um usuário para outro no bate-papo, a mensagem é primeiramente enviada ao servidor, o qual contém uma lista com todos os clientes *online* naquela fase ou atividade, e reenvia a mensagem recebida ao cliente destino da mensagem. No cliente, a classe *ChatInterface* é responsável apenas por realizar a conexão com o servidor e criar algumas *threads* de controle. O bate-papo cliente foi implementado em um esquema conhecido como Produtor-Consumidor, onde existem 4 *thread* que são responsáveis por enviar e receber mensagens do servidor. Para realizar a comunicação entre estas *threads*, existem dois *buffers*, um para enviar e outro para receber. Quando uma mensagem é enviada, a classe *ChatInterface* invoca a *thread EnviarProdutor*, a qual insere a mensagem no *buffer* de envio. A classe *EnviarConsumidor*, por sua vez, espera por novas mensagens no *buffer* para enviar, e quando há uma mensagem, ela retira a mensagem do *buffer* e a envia ao servidor. O *buffer*, que também é uma *thread*, necessita ser sincronizado, ou seja, seus métodos necessitam ser *syncronized* para que quando, por exemplo, o *EnviarConsumidor* verificar que não há novas mensagens no *buffer* para serem enviadas, esta *thread EnviarConsumidor* espere. Quando o *EnviarProdutor* insere uma nova mensagem no *buffer*, este notifica o *EnviarConsumidor* de que uma nova mensagem foi adicionada a ele. As mensagens enviadas tem os atributos origem, destino, mensagem e um booleano que indica se a mensagem é para todos ou não.

No recebimento de novas mensagens, as *threads ReceberProdutor* e *ReceberConsumidor* entram em ação juntamento com um *buffer*. A *thread ReceberProdutor* tem o papel de ficar esperando novas mensagens do servidor através do *socket*. Quando uma nova mensagem chega, esta verifica primeiramente o tipo da mensagem, se é uma mensagem comum, uma atualização da lista de clientes online ou uma atualização de *status* (algum cliente entrou ou saiu). Se for uma mensagem comum, ele recebe a origem, o destino e a mensagem, colocando-os em forma de objeto *Mensagem* no *buffer*, o qual prontamente notifica a *thread ReceberConsumidor* da existência de uma nova mensagem. Esta *thread* apenas retira a mensagem do *buffer* e a apresenta na interface para o usuário.

Se a mensagem recebida for uma atualização da lista de clientes, os nomes dos clientes são recebidos um a um e diretamente mostrados na lista de clientes para o usuário através de um *JComboBox*. Por fim, se a mensagem recebida é uma atualização de status, ela é colocada no *buffer* assim como uma mensagem comum. Desta forma, o funcionamento do bate-papo do módulo cliente foi descrito.

No lado do servidor, a *thread EsperaConexoesChat* fica a espera de novos clientes naquela porta. Quando um cliente requer uma conexão, a *EsperaConexoesChat* a aceita e

recebe o nome do cliente, iniciando uma *thread ClienteChat* assim como na parte principal da apliação servidora. Após a inicialização da *thread*, ainda na *EsperaConexoesChat*, todos os usuários são avisados sobre a entrada de um novo cliente na sala, percorrendo um *ArrayList* que contém as *threads ClienteChat* de todos e enviando a mensagem. Também é realizada a atualização da lista de clientes para todos, enviando a lista de clientes atualizada para todos os clientes ativos. Como a função do servidor de bate-papo é basicamente repassar mensagens, mantendo uma lista de clientes ativos, as *threads ClienteChat* ficam a espera de novas mensagens dos clientes. Quando uma mensagem é recebida, esta é montada em um objeto *Mensagem* e enviada ao método *enviarMsgPara*, o qual verifica se a mensagem recebida tem como destino todos os usuários conectados ou apenas um. Se for apenas um, a *thread ClienteChat* do cliente destino é recuperada e utilizada o método *enviaMsg*, o qual envia o tipo da mensagem e a mensagem ao cliente. Se for para todos, o *ArrayList* de clientes ativos é percorrido, enviando a mensagem para cada um. Desta forma, a parte servidora do bate-papo foi explanada.

O processo de início de um bate-papo em uma fase é ilustrado pelos diagrama de sequência da Figura 21 e diagrama de colaboração contido na Figura 22.

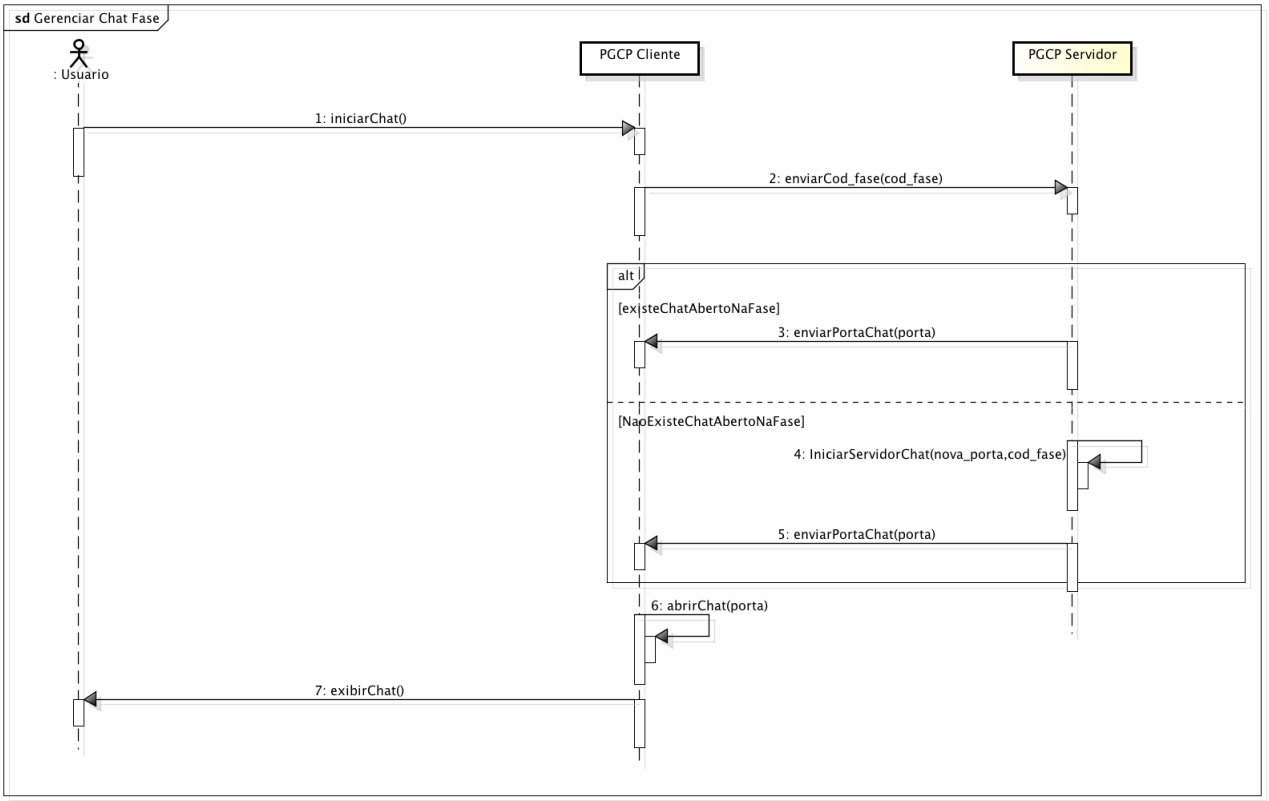

Figura 21: DSS Iniciar chat em uma Fase

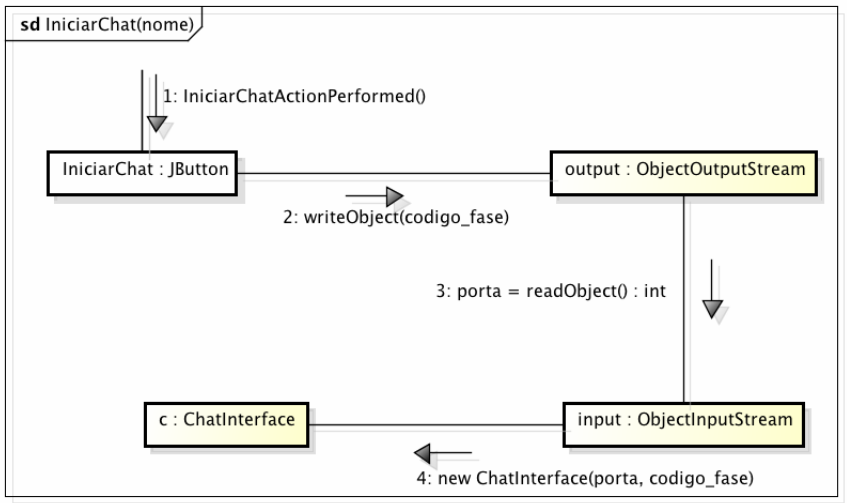

Figura 22: Diagrama de Colaboração - Iniciar chat em uma Fase

Quando um usuário deseja sair do bate-papo, este envia uma mensagem utilizando o *socket* da aplicação, e não do bate-papo, enviando o código da fase na qual o bato-papo será fechado. O módulo cliente então fecha todas as *thread* do Produtor-Consumidor ativas. No servidor da aplicação, o objeto *ChatPorta* é excluído do *ArrayList ChatFase*, indicando que o cliente não está mais online naquela fase. O servidor então verifica se o cliente que está saindo é o último cliente ativo na sala. Se for, o método *CloseServer* da classe *ServidoChat* é invocado, o qual destroe a *thread* do cliente ativo e invoca um método do *EsperaConexoesChat* que irá inserir na base de dados uma nova ocorrência de um bate-papo realizado. Esta *thread* então também é fechada e por fim o *ServerSocket* deste chat é finalizado. Por outro lado, se o usuário requisitando sua saída não for o último, como o *socket* do chat na aplicação cliente foi fechado, a *thread ClienteChat*, no momento em que for esperar uma nova mensagem, verificará que o *socket* foi fechado, gerando uma exceção que irá remover o cliente, retirando-o da lista de clientes, atualizando a lista de clientes de todos os clientes ativos e avisando a saída deste cliente através de uma mensagem.

3.1.3 Repositório de Arquivos

Na PGCP, usuários podem fazer *upload* de seus arquivos em fases, compartilhando-os com outros usuários, os quais podem fazer *download* destes arquivos. Portanto existem duas ações distintas: o *upload* de um arquivo e o *download*. É válido registrar que apenas administradores (de projeto e de sistema) e o usuário que realizou o *upload* de um arquivo tem permissão para apagá-lo.

Iniciando com a ação de *upload*, quando o usuário faz a operação de enviar um arquivo

ao servidor, o módulo cliente utiliza o *socket* para enviar um código ao módulo servidor, o qual indica que aquele deseja realizar uma operação sobre arquivos. Posteriormente, o cliente envia um código que identifica o tipo de operação, *upload* ou *download*. Como há um repositório de arquivos para cada fase de um projeto, o cliente então envia o código da fase para o servidor. O servidor por sua vez, recebe estes dados, procura uma porta livre, a envia ao cliente e abre uma instância da classe *ServidorArquivo*, passando como parâmentro a porta, o código da fase e o tipo de operação (receber arquivo). Do lado do cliente, a porta é recebida e uma instância da classe *ClienteArquivo* é aberta. A partir daí, assim como no bate-papo, um novo *ServerSocket* é utilizado. Este *ServerSocket* é criado pelo *ServidorArquivo* e fica aguardando alguém se conectar a ele. Como no caso da tranferência de arquivos não há mais de um usuário conectado ao mesmo tempo, não é necessário a criação de *threads* que controlem esta concorrência. O cliente então realiza a conexão a este novo *ServerSocket*, abre uma instância da classe *JFileChooser* para o usuário escolher o arquivo a ser enviado, envia o nome deste arquivo ao servidor e o transfere *byte* a *byte*. No servidor, o nome do arquivo é recebido, o arquivo é recebido *byte* a *byte* e ele é colocado sempre na raiz do módulo servidor. Após a tranferência, o módulo cliente cria um novo registro na base de dados, relatando o envio deste arquivo para que outros usuários possam realizar o *download* dele posteriormente.

O processo de envio de um arquivo em uma fase é ilustrado pelos diagramas de sequência da Figura 25 e diagrama de colaboração contido na Figura 24.

No *download* do arquivo por parte do cliente, a diferença é que o cliente não envia o código da fase ao servidor, e sim diretamente o código do arquivo a ser baixado, que está contido na base de dados. O servidor então cria novamente uma instância do *ServidorArquivo*, mas desta vez chamando o método *enviarAquivo*, o qual faz uma busca na base de dados pelo arquivo com o código recebido, obtém o nome deste arquivo, carrega-o em um objeto do tipo *File* e o envia *byte* a *byte* ao cliente. O cliente, por sua vez recebe o nome do arquivo, um *JFileChooser* é aberto para que o usuário escolha onde irá salvar o arquivo, e este é então recebido.

Este processo de *download* de um arquivo em uma fase é ilustrado pelo diagrama de sequência contido na Figura 25.

3.1.4 Áudio e Videoconferência

A audioconferência e videoconferência são muito semelhantes no seu funcionamento, ao passo que só é explicada a audioconferência. Ao final da explicação, as pequenas

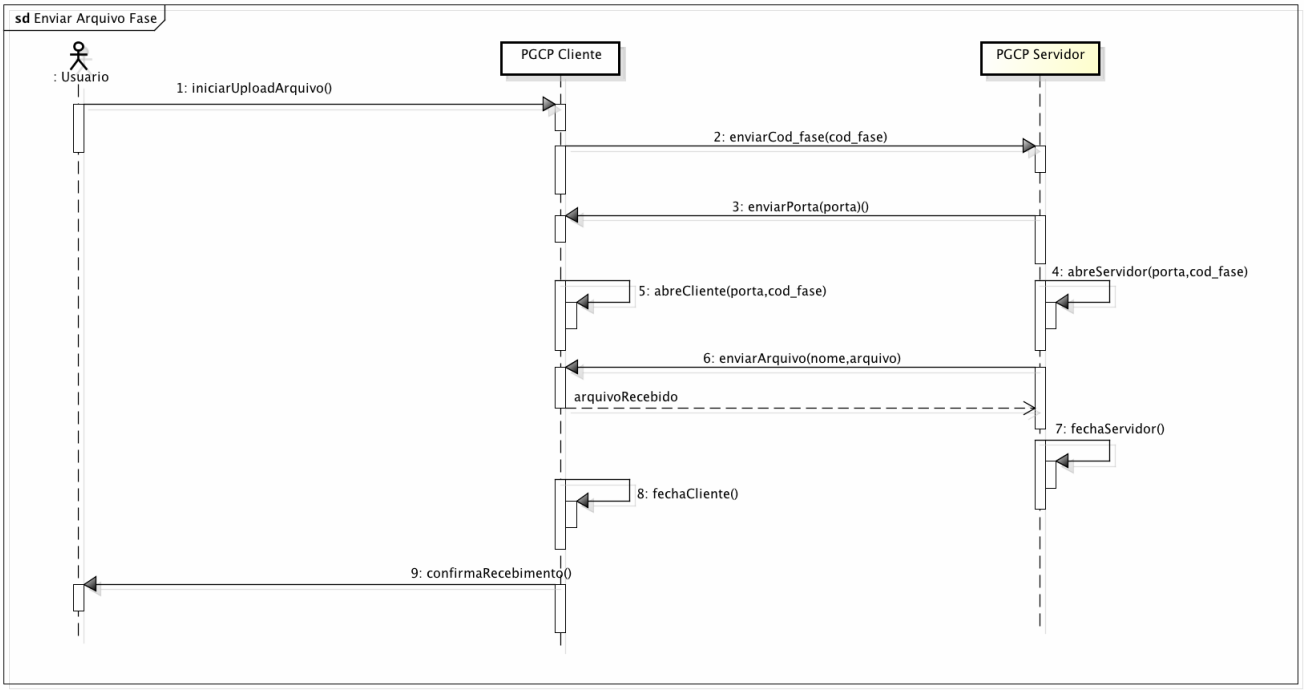

Figura 23: DSS Upload Arquivo em uma Fase

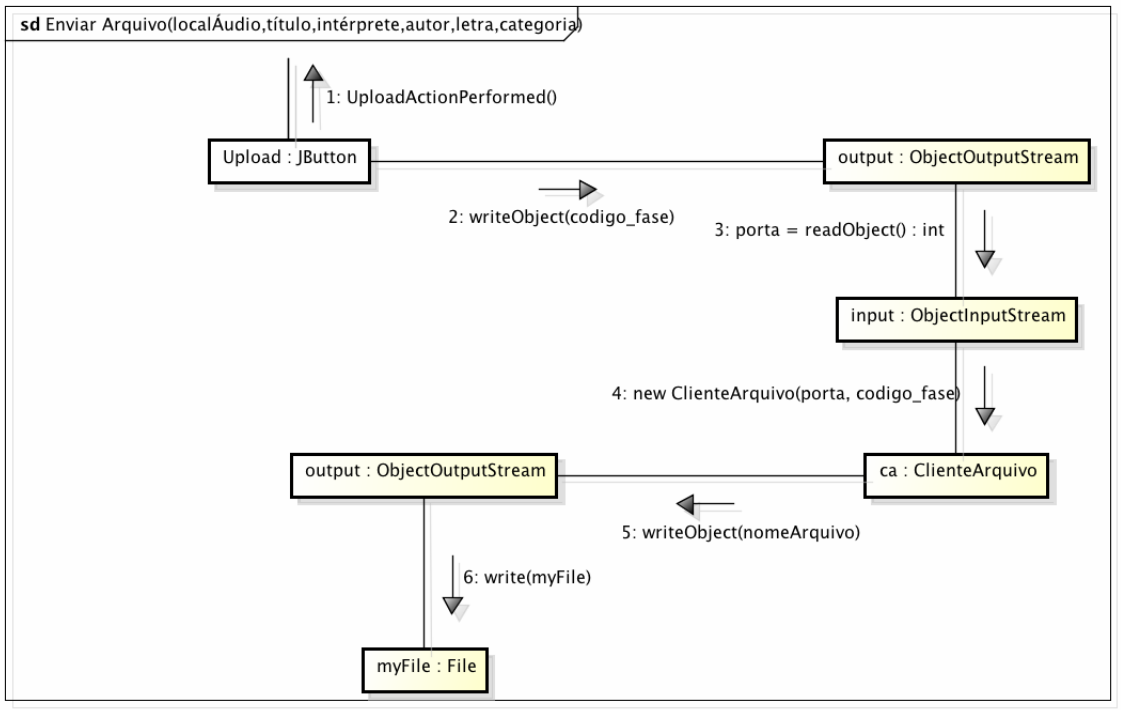

Figura 24: Diagrama de Colaboração - Upload Arquivo em uma Fase

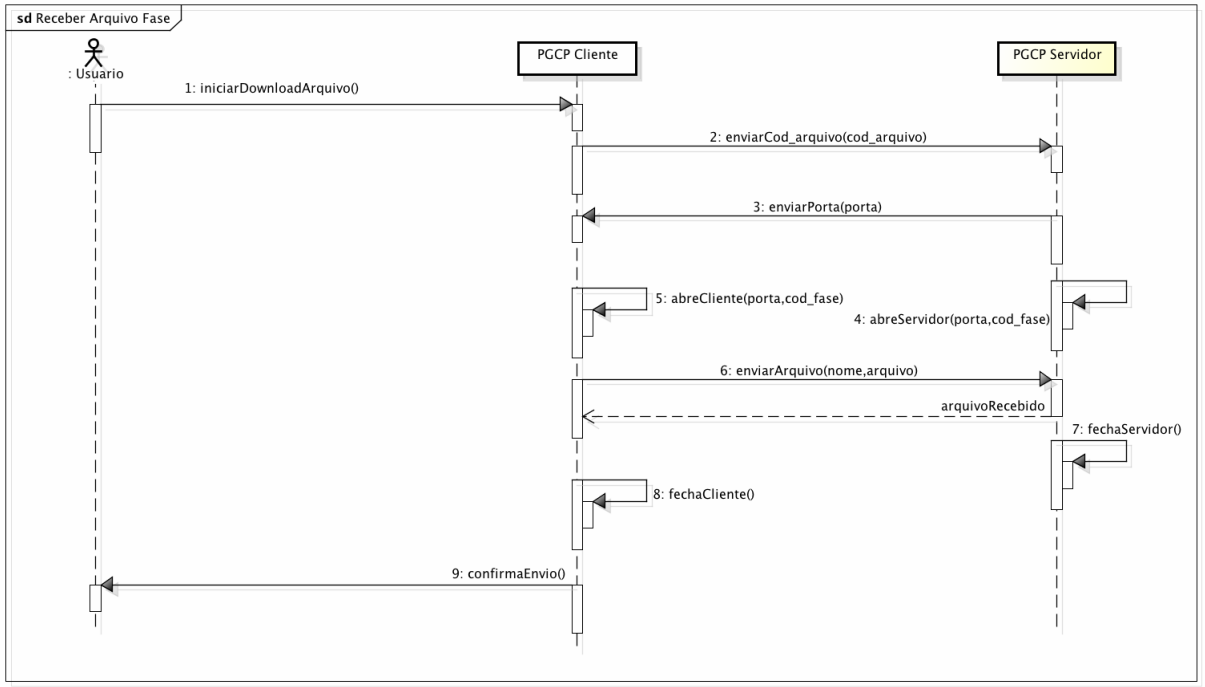

Figura 25: DSS Download Arquivo em uma Fase

diferenças entre uma e outra ferramenta são listadas.

Assim como o bate-papo e o repositório de documentos, a audioconferencia utiliza o envio de códigos (todos eles números inteiros) para informar o servidor sobre a intenção de iniciar uma nova instância. Inicialmente, o cliente envia ao servidor dois códigos, o primeiro indicando que se trata de audioconferência e o segundo que a intenção é de iniciar uma nova. No servidor, os códigos são recebidos e é verificado se já existe alguma audio ou videoconferência aberta naquele cliente, pois ao contrário do bate-papo, o usuário não pode utilizar estes recursos mais de uma vez ao mesmo tempo. Se já houver alguma aberta, o cliente é avisado e nada é feito. Se não houver, o cliente é liberado para proceder com as requisições. Este então envia ao servidor um código que indica se a audioconferência será usada em uma fase ou atividade, enviando junto o código desta. Assumindo que será iniciada em uma fase, o servidor percorre o *ArrayList* que contém todos os clientes conectados a ele procurando por emissores naquela fase.

Se não for encontrado nenhum emissor, o servidor inclui uma audioconferência na base de dados para registrar o início de uma. Por fim o servidor envia ao cliente uma porta livre para realizar a transmissão e o código da audioconferência adicionada na base de dados. O cliente então recebe estas informações, inicia um emissor de áudio naquela porta e realiza um *update* no registro da audioconferência na base de dados, adicionando

o nome dele mesmo como um dos participantes. Após isto, é iniciada uma *thread* no cliente, a qual fica a espera por mensagens do servidor indicando novos emissores naquela fase, para que este cliente abra um receptor para aquele novo emissor.

Se, quando o cliente for iniciar a audioconferência, já houver algum emissor naquela fase, o servidor envia ao cliente uma porta livre para que ele possa iniciar seu próprio emissor. Envia também o código da audioconferência na base de dados, e a cada cliente que estiver emitindo naquela fase, envia o IP, a porta e o nome do emissor para o cliente que irá iniciar a emissão. Por fim, o servidor envia os mesmos atributos (IP, porta e nome) deste cliente para todos os emissores, para que estes possam iniciar receptores em suas instâncias.

O emissor utiliza algumas classes como *Processor*, *MediaLocator*, *TrackControl*, *CaptureDeviceInfo*, *DataSource* e *DataSink* para realizar a transmissão. Tais classes fazem parte do *framework* JMF, o qual já foi descrito na seção 2.3.1. Para realizar a emissão para mais de um IP ao mesmo tempo, ou seja, para que mais de uma pessoa possam ouvir o que outra está falando, é necessário que no *MediaLocator* do emissor seja informado um IP de *broadcast*.

O receptor utiliza apenas um *MediaLocator* e um *Player*, sendo o *MediaLocator* o responsável por indicar o IP e a porta ao qual o *Player* irá se conectar. Portanto para realizar a emissão, o IP necessita ser de *broadcast*, porém todos os receptores devem conhecer o IP de cada transmissor. Desta forma, faz-se necessário que o servidor mantenha uma lista com o IP de todos os clientes ativos no sistema.

O processo de inicar uma audioconferência em uma fase é ilustrado pelos diagramas de sequência da Figura 26 e diagrama de colaboração contido na Figura 27.

Quando um cliente requisita a desconexão a uma audioconferência, ele envia um código ao servidor que identifica o fechamento da audioconferência. O cliente então fecha o emissor através do método *fechar* da classe *EmissorAudio*, o qual para o *DataSink* e o *Processor*. Também é chamado o método *fechar* da *ThreadEsperaReceptor* (a qual espera novos emissores), o qual percorre o *ArrayList* de receptores abertos e os fecha um a um. Esta *thread* então é parada e finalizada. O servidor, ao receber o aviso de desconexão, percorre todos os clientes que estão transmitindo naquela fase, enviando o IP, porta e nome do emissor desconectado. Cada cliente conectado, através da *ThreadEsperaReceptor*, recebe estas informações, localiza o emissor desconectado no *ArrayList* de receptores e o remove.

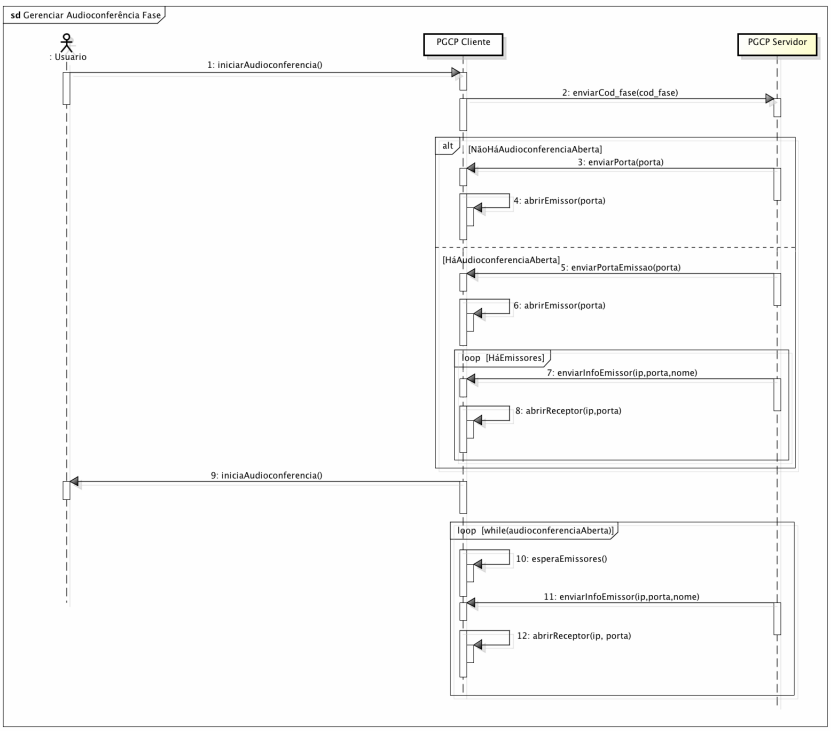

Figura 26: DSS Iniciar Audioconferência em uma Fase

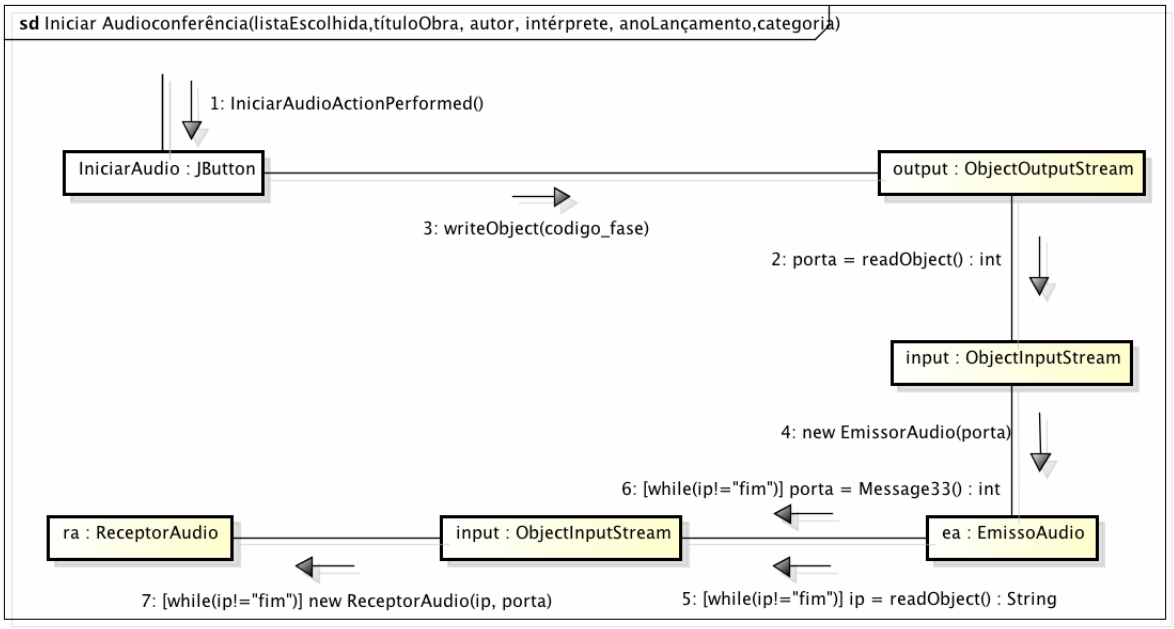

Figura 27: Diagrama de Colaboração - Iniciar Audioconferência em uma Fase

A diferença da audioconferência para a videoconferência é que, na videoconferência, é necessário abrir uma audioconferência também, para que sejam transmitidos imagem e áudio. Na implementação do emissor e receptor, a diferença é na captura do dispositivo de emissão e no *MediaLocator*, que deve conter ao final da *url*, a palavra "video"no lugar de "audio"da audioconferência, tanto na emissão quanto na recepção.

O processo de inicar uma videoconferência em uma fase é muito semelhante a audioconferência e é ilustrado pelo diagrama de sequência contido na Figura 28.

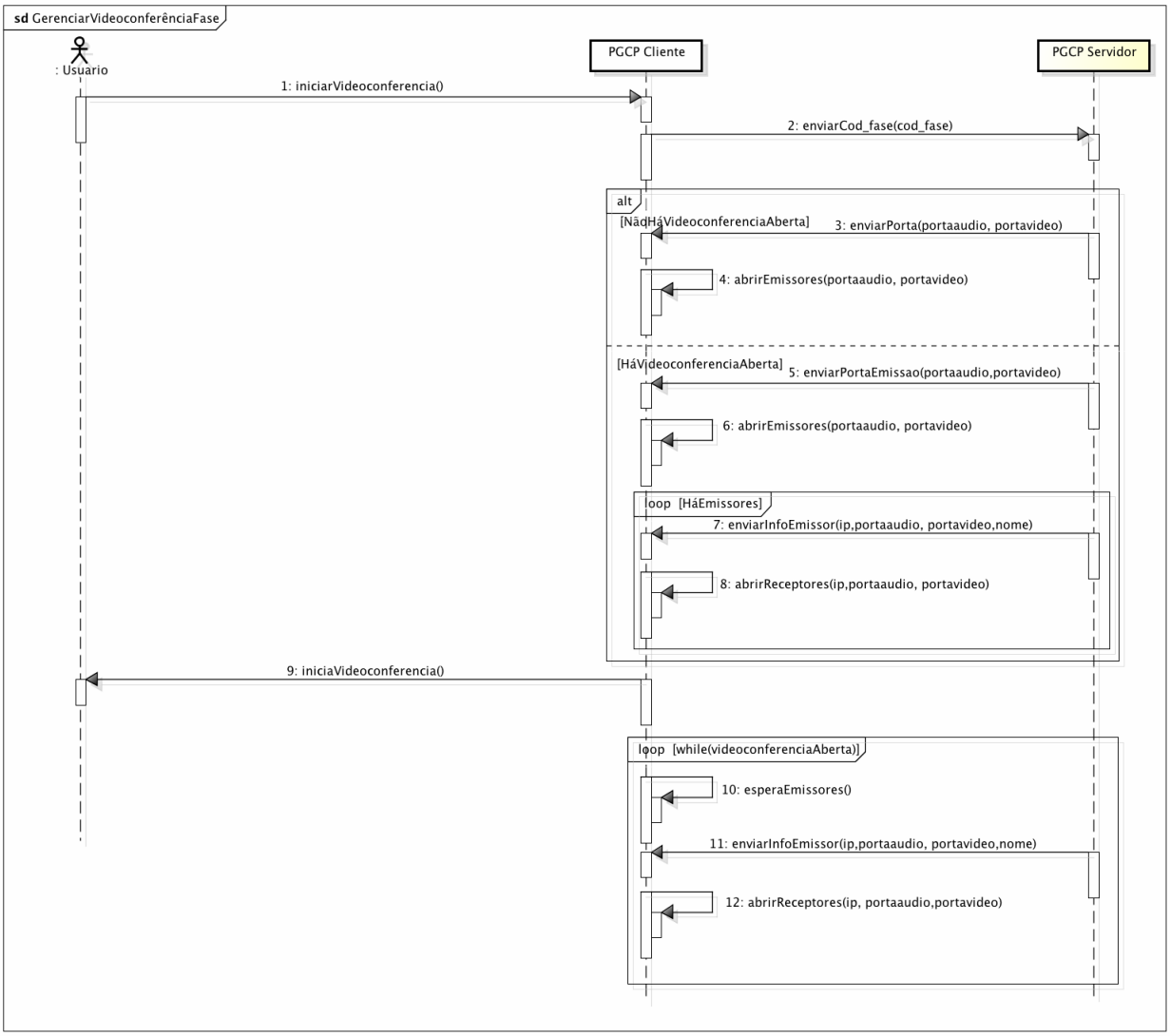

Figura 28: DSS Iniciar Videoconferência em uma Fase

Capítulo 4

Estudo de Caso - Sistema Proex

4.1 Introdução

Neste capítulo é apresentado um estudo de caso utilizando a PGCP. Para realizá-lo, foram introduzidas informações de um sistema em desenvolvimento da Unesp chamado Sistema PROEX (Pró-Reitoria de Extensão Universitária).

Este sistema está sendo desenvolvido na Faculdade de Ciências e Tecnologia - FCT - da Unesp de Presidente Prudente pelos funcionários do Serviço Técnico de Informática (STI).

O Sistema PROEX é um ambiente para gerenciar projetos acadêmicos. A PROEX é um órgão da UNESP que tem o compromisso com o desenvolvimento das atividades de Extensão Universitária.

Já existe um sistema em atividade, porém ele não atende às necessidades atuais. Desta forma, iniciou-se o desenvolvimento de um novo sistema.

4.2 Sistema PROEX na PGCP

Nesta seção é apresentada a interface da PGCP, introduzindo os dados do desenvolvimento do Sistema PROEX fornecidos pelos desenvolvedores.

Para que se tenha uma visão geral da PGCP, a Figura 29 ilustra o fluxo de operações que os usuários podem realizar dentro da PGCP.

Foram criados usuários para todos os três funcionários que estão desenvolvendo o

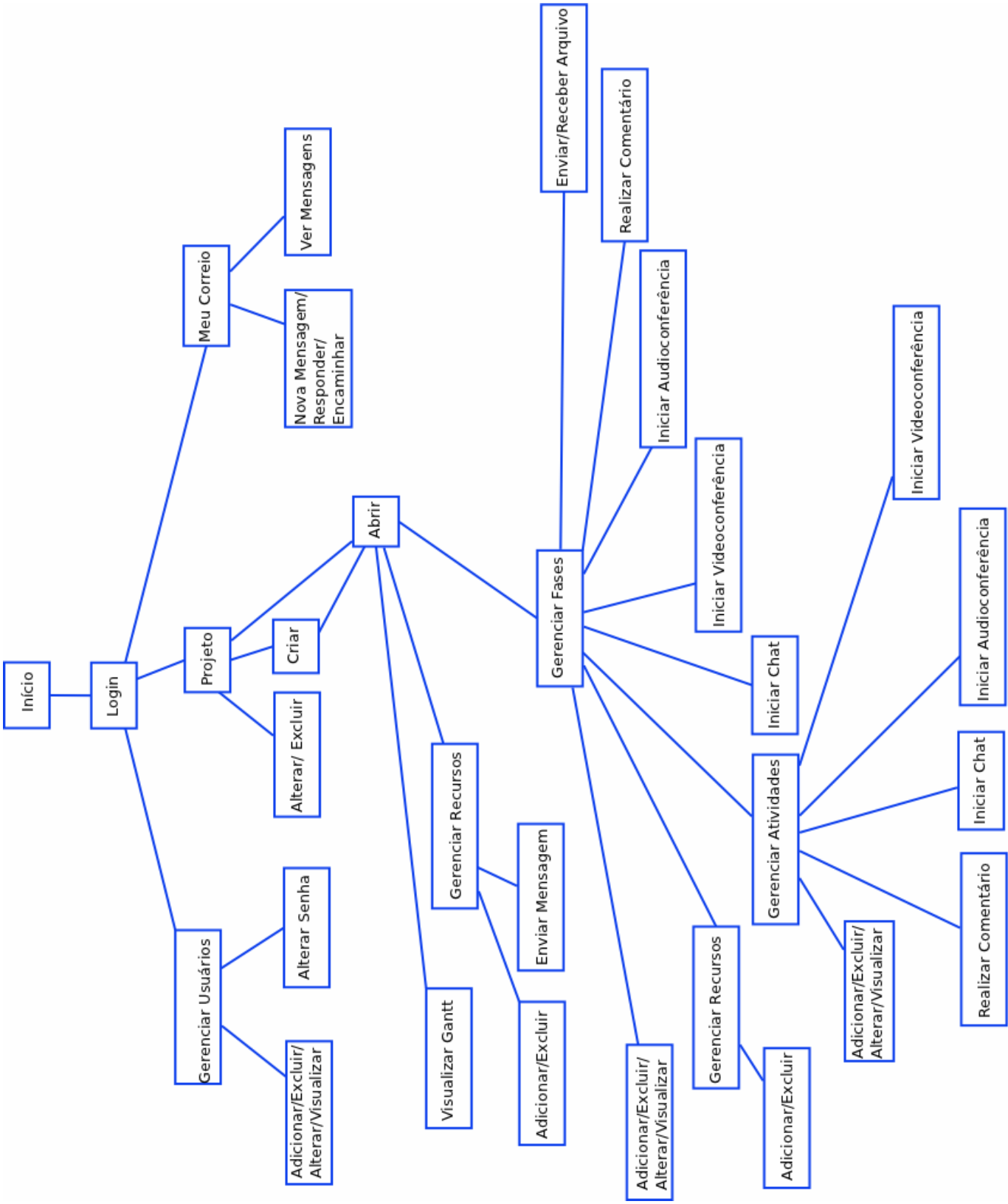

Figura 29: Fluxo de operações do usuário.

sistema, e também para outros três bolsistas, os quais auxiliam no desenvolvimento. Um usuário dos três funcionários foi criado como administrador do sistema, para que este criasse os demais usuários do projeto, para a criação do projeto e para a inclusão dos dados.

Na Figura 30 é ilustrada a interface principal da PGCP, com um usuário administrador de projeto já autenticado.

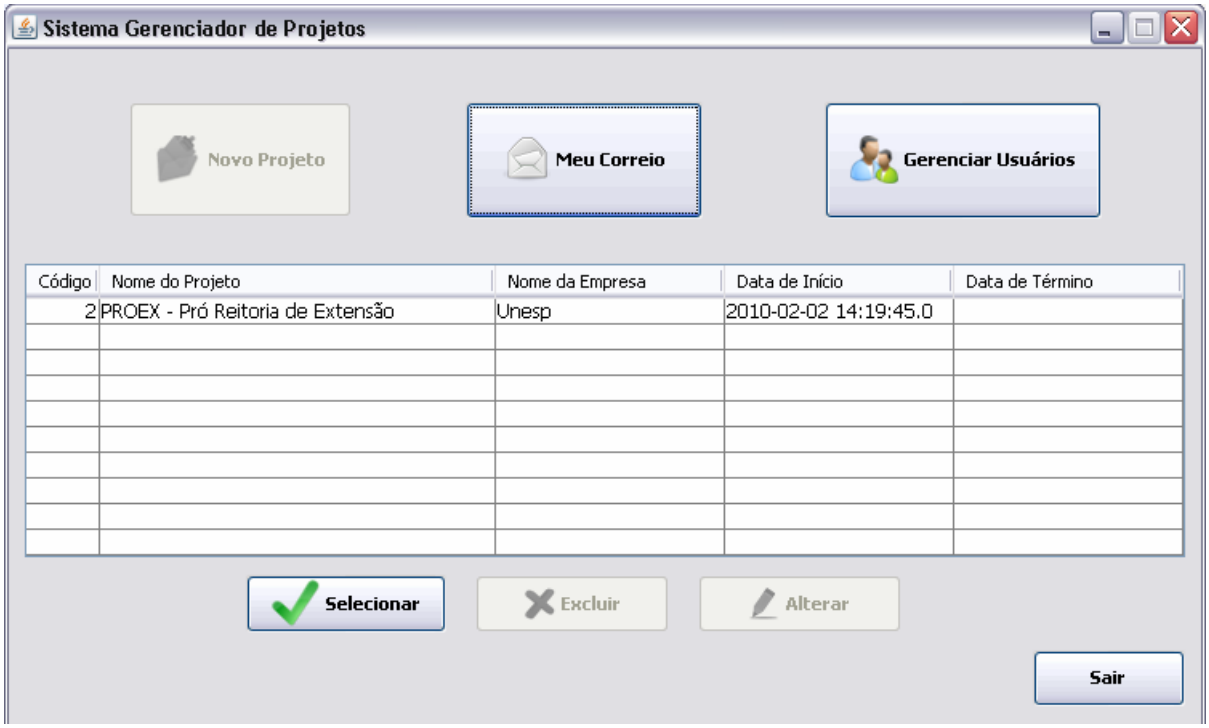

Figura 30: Interface Principal.

Na Figura 31 é ilustrada a interface de alteração do projeto, onde é possível visualizar os dados do projeto e alterá-los.

Na Figura 31, pode-se observar que a data de término do projeto não foi informada, isto devido ao fato de o projeto ainda estar em desenvolvimento.

Quando o projeto é aberto, selecionando o projeto na interface principal e escolhendo a opção Selecionar, a interface principal de projeto é aberta, conforme é ilustrado na Figura 32.

Pode-se notar que a interface é semelhante ao programa Planner apresentado na seção 2.3.5. Do lado esquerdo existem três botões principais. Quando a interface de projeto é aberta, a área de visualização do gráfico de gannt e uma tabela contendo todas as fases adicionadas são mostradas. Nesta área, é possível acompanhar o cronograma das fases através de um gráfico de barras seguindo uma linha do tempo. Também é possível alterar

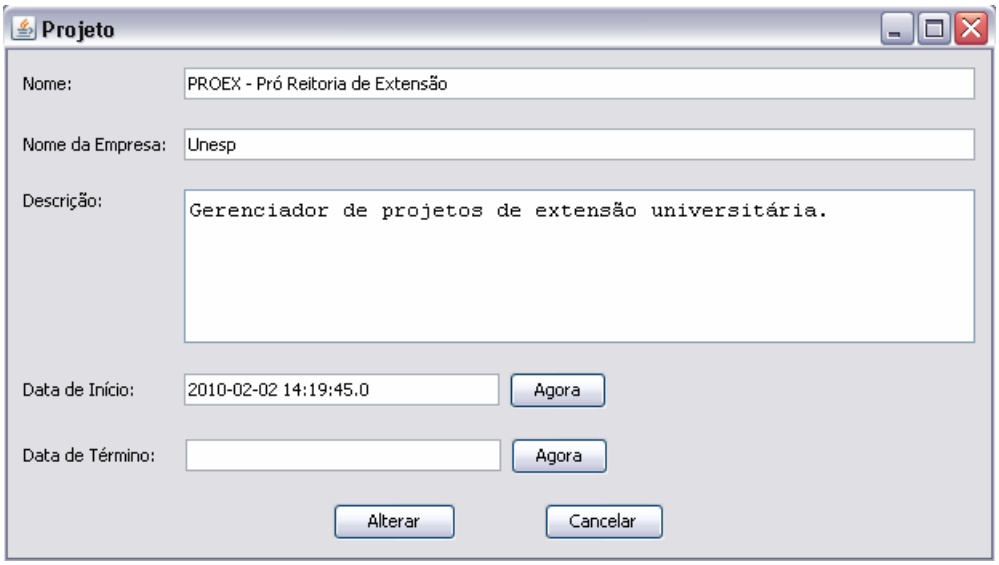

Figura 31: Interface de alteração de Projeto.

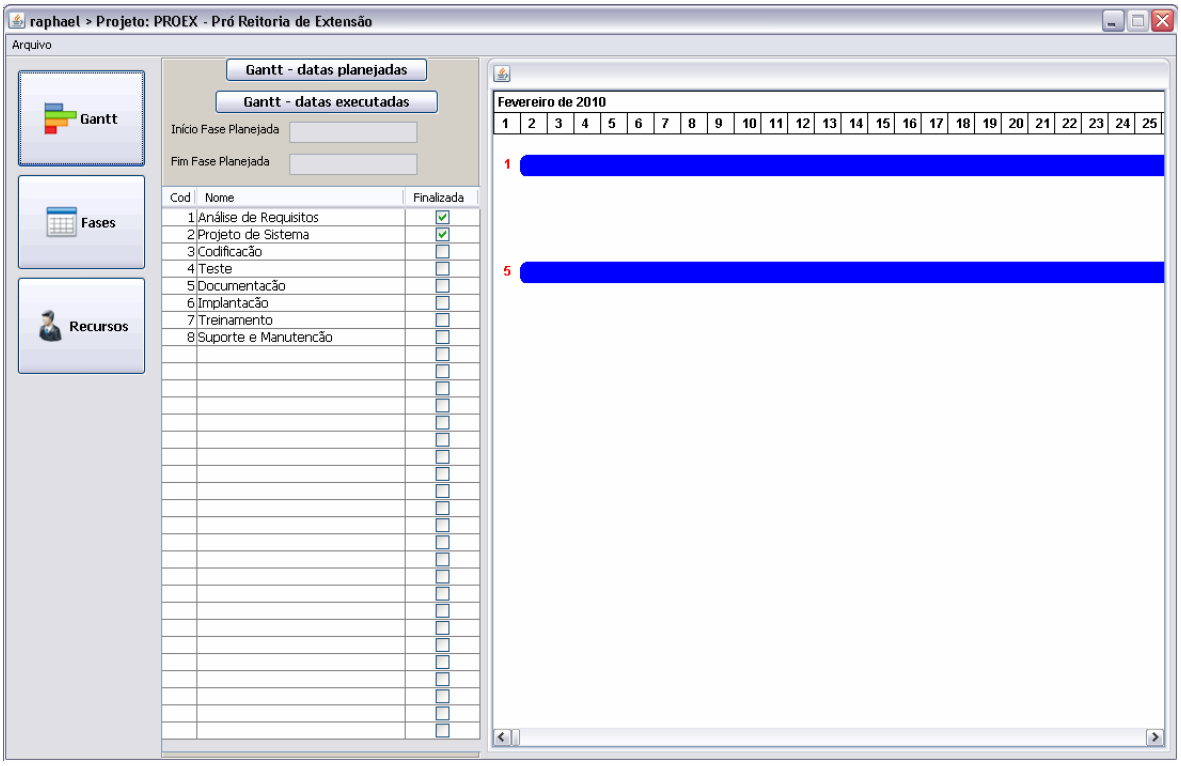

Figura 32: Interface principal de Projeto.

entre a data planejada e executa das fases do projeto.

Pode-se visualizar neste caso, que as fases de Análise de Requisitos e Projeto de Sistema já estão finalizadas. Selecionando-se cada fase da tabela, os atributos acima da tabela são preenchidos com as datas, auxiliando na visualização do gráfico de Gantt.

A opção Recursos leva à área onde são visualizados os recursos do projeto, adicionados novos recursos e onde também é possível atribuir o privilégio de administrador a um usuário. Esta área é ilustrada pela figura 33.

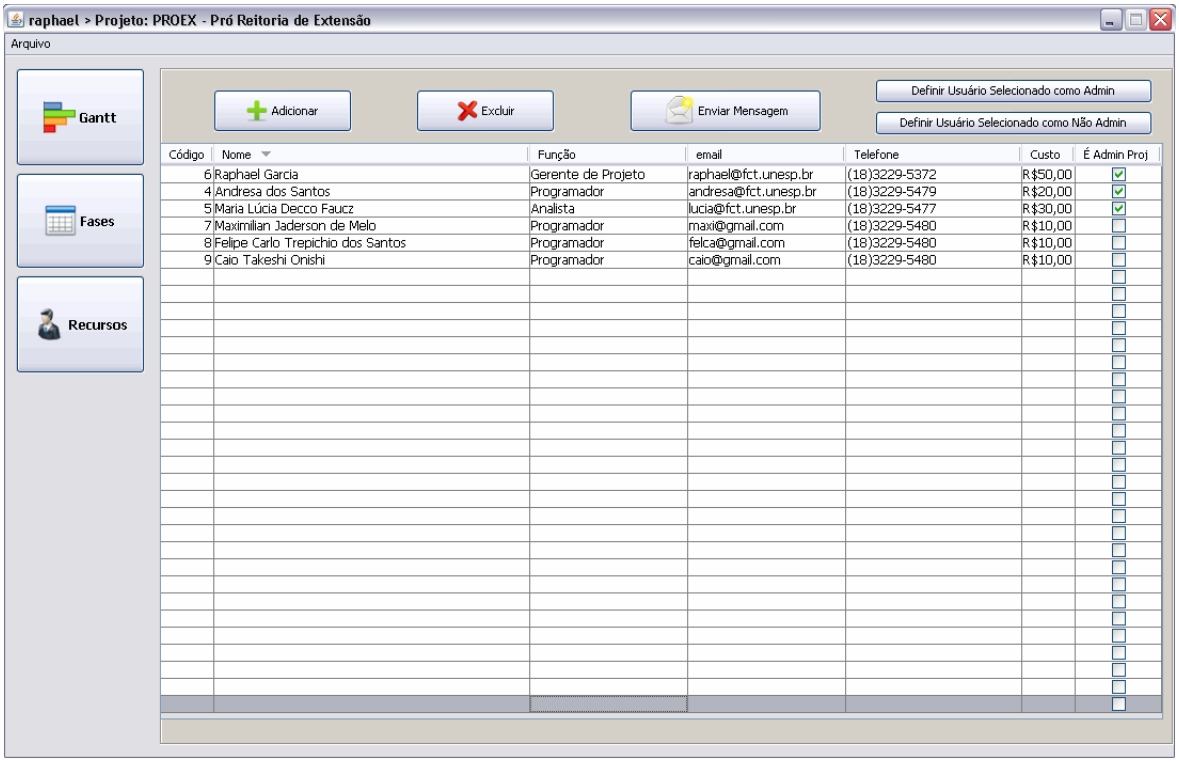

Figura 33: Interface Recursos do Projeto.

Pode-se obervar que três usuários possuem privilégios de administrados do projeto, enquanto os demais usuários não possuem tal privilégio. Através desta interface, também é possível enviar uma mensagem a um usuário utilizando o correio eletrônico da PGCP. A interface do correio eletrônico será abordada no final desta seção.

Escolhendo-se a operação de adicionar, o qual só é habilitado para usuários administradores do projeto, a interface contida na Figura 34 é exibida.

Nesta interface, todos os usuários do sistema são exibidos, e para adicionar usuários ao projeto, basta selecioná-lo e escolher a opção Adicionar um a um. Quando não houver mais usuários a ser adicionados, a opção Finalizar adiciona todos os usuários ao projeto.

A opção Fases do canto esquerdo da interface de projeto leva à área com maior quan-

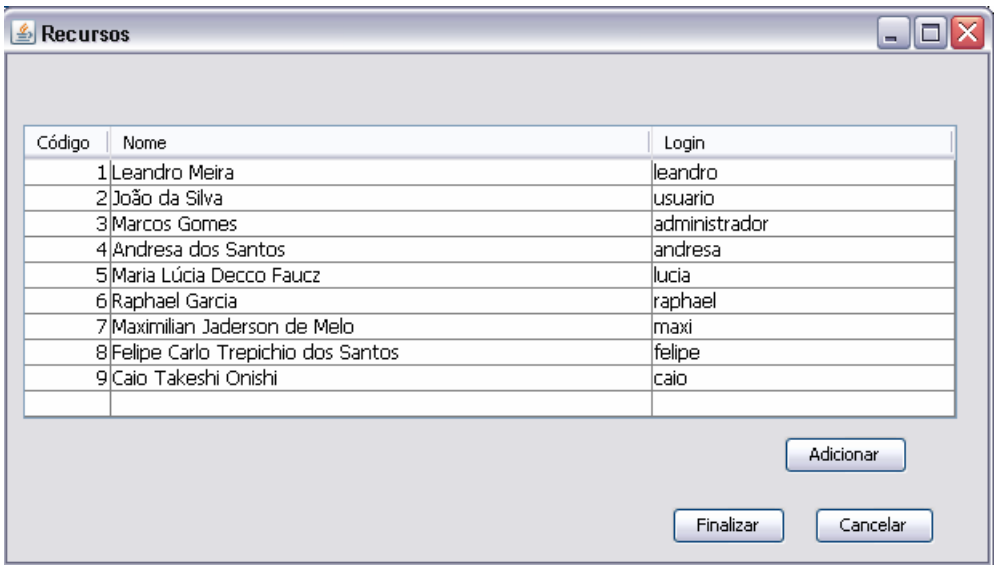

Figura 34: Interface Adicionar Recursos do Projeto.

tidade de recursos da PGCP, como é ilustrada na Figura 35.

Nesta área pode-se visualizar as fases adicionadas ao projeto juntamente com seus prazos, criador e um atributo boleano que indica se a fase está finalizada ou não. Para adicionar uma fase, basta escolher a opção Adicionar, o qual apresenta a interface contida na Figura 36.

São requisitados atributos como nome, descrição e datas planejadas para início e término da fase. Para inserir as datas executadas de uma fase o usuário deve selecionar uma fase e alterá-la.

Cada fase deve conter recursos atribuídos, os quais indicam que estes recursos serão responsáveis por desenvolvê-la. Esta operação é possível através da seleção de uma fase e a escolha da opção Gerenciar Recursos da Fase, o qual abre a interface ilustrada pela Figura 37.

Esta área trabalha com um esquema de adicionar e retirar um recurso da fase selecionada. No caso da fase selecionada, a fase de Codificação, a tabela de baixo mostra os usuários adicionados a esta fase e a tabela acima mostra todos os recursos incluídos no projeto. Pode-se observar que existem recursos do projeto que não estão atribuídos a esta fase. Se quiser adicionar um novo recurso à fase, basta selecioná-lo na tabela de cima e escolher a opção Adicionar. Para retirá-lo da fase, basta fazer o processo inverso, selecionando o recurso a ser removido e escolhendo a opção Retirar.

Como se sabe, cada fase pode conter atividades incluídas. Tomando como exemplo a fase de Projeto de Sistema, selecionando-a e escolhendo a opção Gerenciar Atividades da

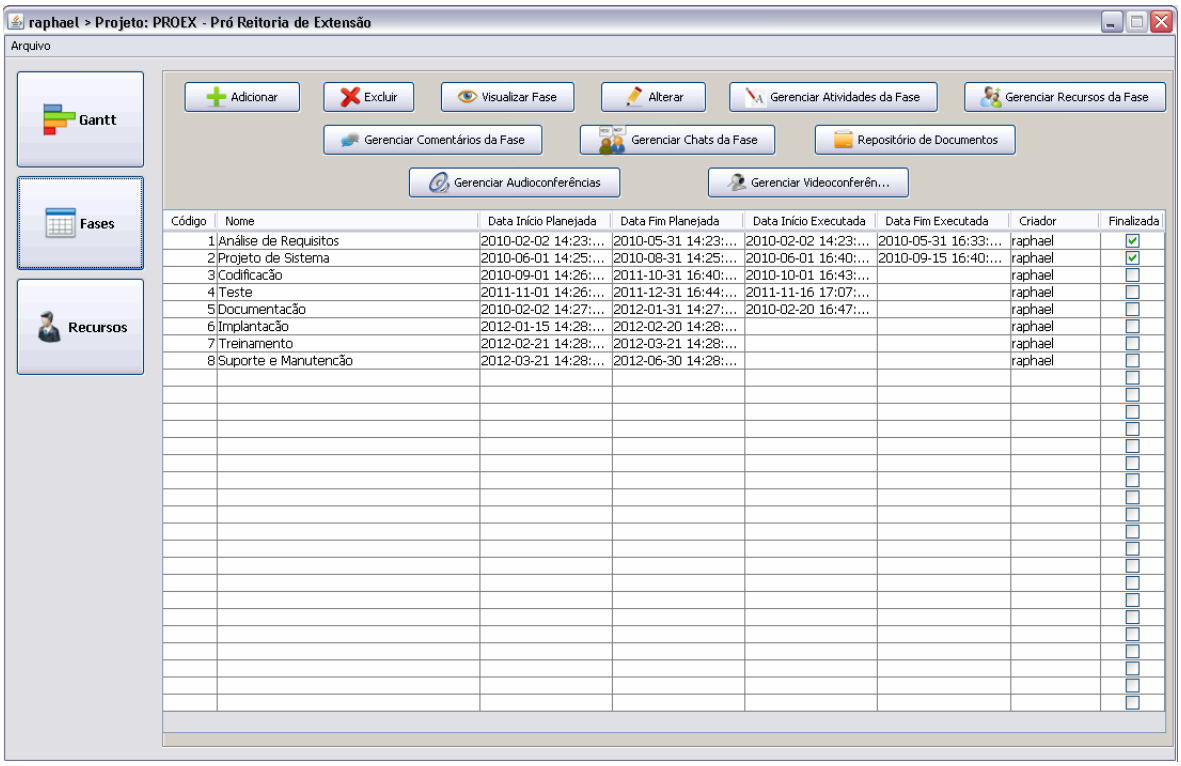

Figura 35: Interface Fases.

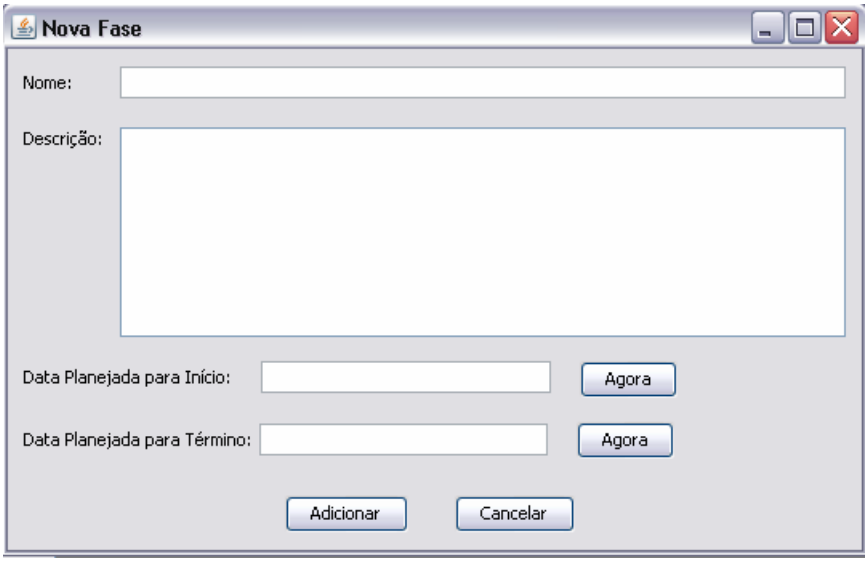

Figura 36: Interface Adicionar Fase.

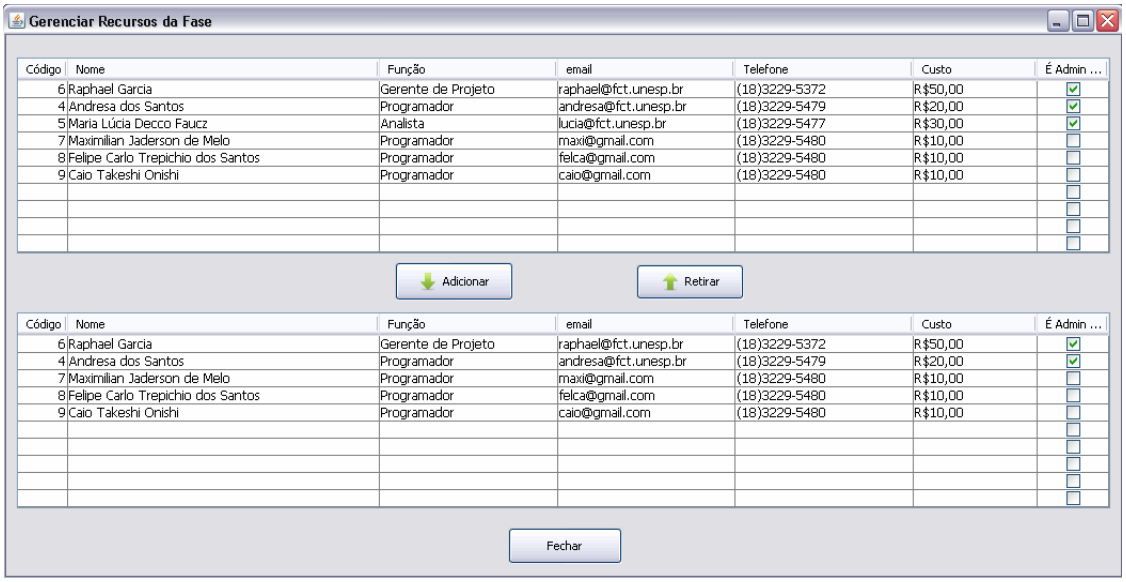

Figura 37: Interface Adicionar Recursos a uma Fase.

Fase, as atividades adicionadas a esta fase são apresentadas, como ilustra a Figura 38.

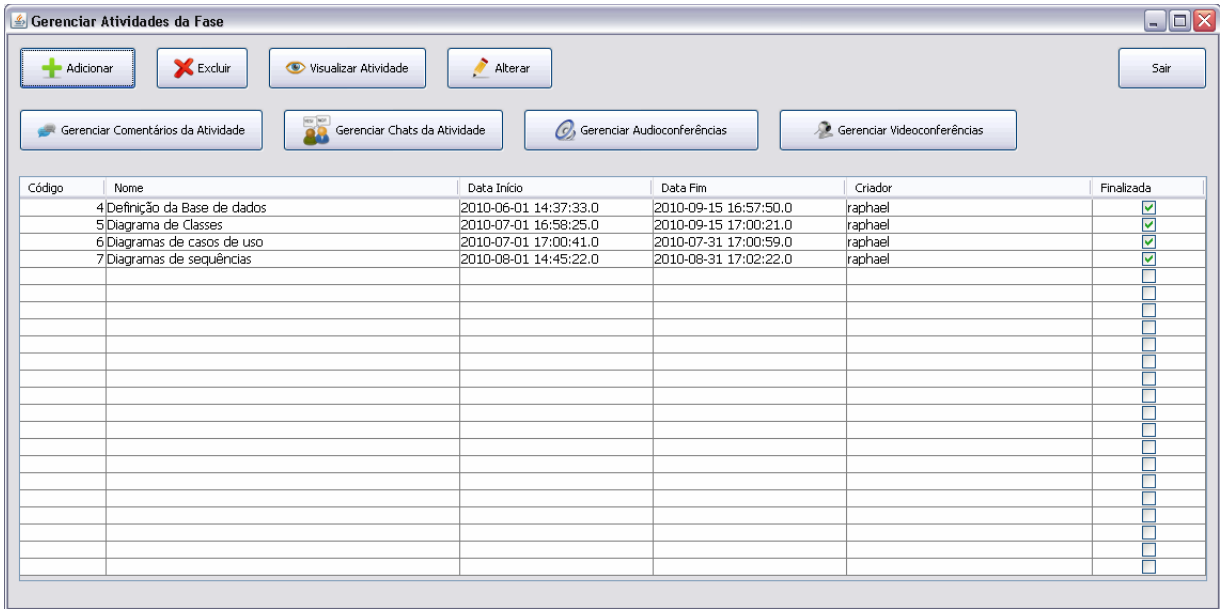

Figura 38: Interface Gerenciar Atividades da Fase.

Pode-se observar as atividades definidas para a fase juntamente com seus atributos. As funções que as opções na parte superior realizam não serão comentadas, pois as operações de adicionar, excluir, visualizar e alterar uma atividade funcionam da mesma forma da fase, e as funções de comentários, chats, audio e videoconferência estão também presentes nas fases, onde elas serão abordadas.

Na interface de visualização das fases, há a possibilidade de adicionar comentários

sobre cada fase, para tanto, basta selecionar uma fase e escolher a opção Gerenciar Comentários da Fase, o qual leva à área ilustrada na Figura 39.

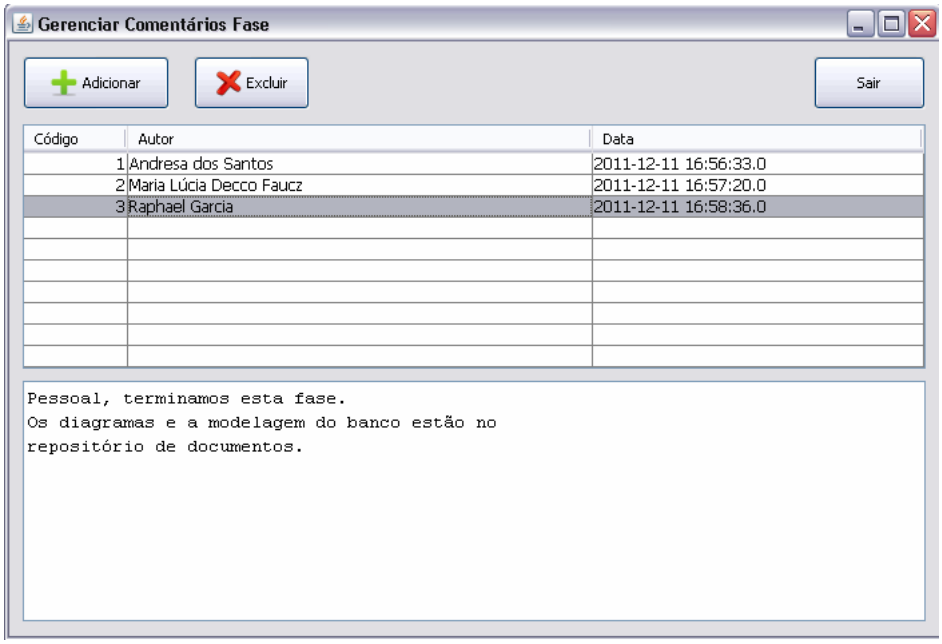

Figura 39: Interface Gerenciar Comentários da Fase.

Nesta área pode-se observar os comentários feitos por outros participantes da fase e também incluir novos comentários. Apenas o autor do comentário ou um administrador podem apagar um comentário.

Um recurso interessante e importante do projeto é o Chat (bate-papo). Ele é acionado através da opção Gerenciar Chats da Fase, que leva à interface representada pela Figura 40.

Na interface que se abre, é possível visualizar quando aconteceram os chats nesta fase, apresentando a data e os participantes. Iniciando um novo chat, é possível conversar com outros integrantes do projeto como ilustra a Figura 41.

Outro recurso colaborativo que a PGCP possui é o repositório de documentos, onde pode existir um repositório por fase. A interface representada pela Figura 42 ilustra a interface e alguns documentos incluídos.

Escolhendo a opção Upload, o usuário seleciona o arquivo em seu computador, o qual será enviado ao servidor. Para baixar um arquivo compartilhado, basta selecionar o arquivo e e escolher a opção Download, escolhendo o local onde será salvo.

A PGCP utiliza dispositivos de entrada e saída de mídia para realizar áudio e videoconferências. Usuários podem conversar apenas com áudio, ou támbém com vídeo. A

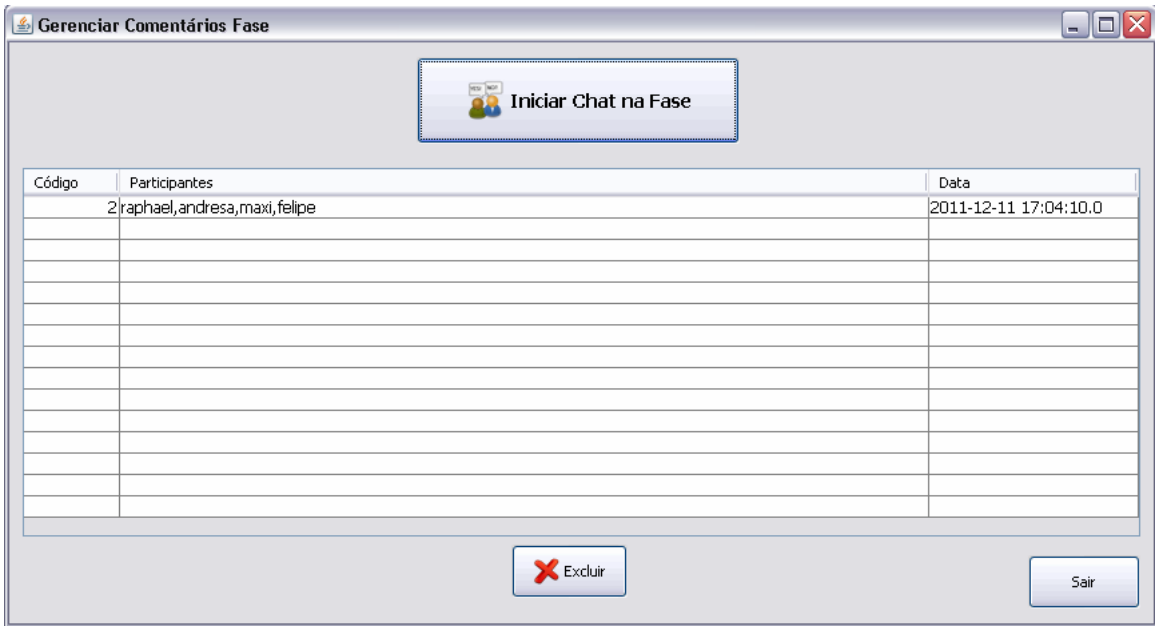

Figura 40: Interface Gerenciar Chats da Fase.

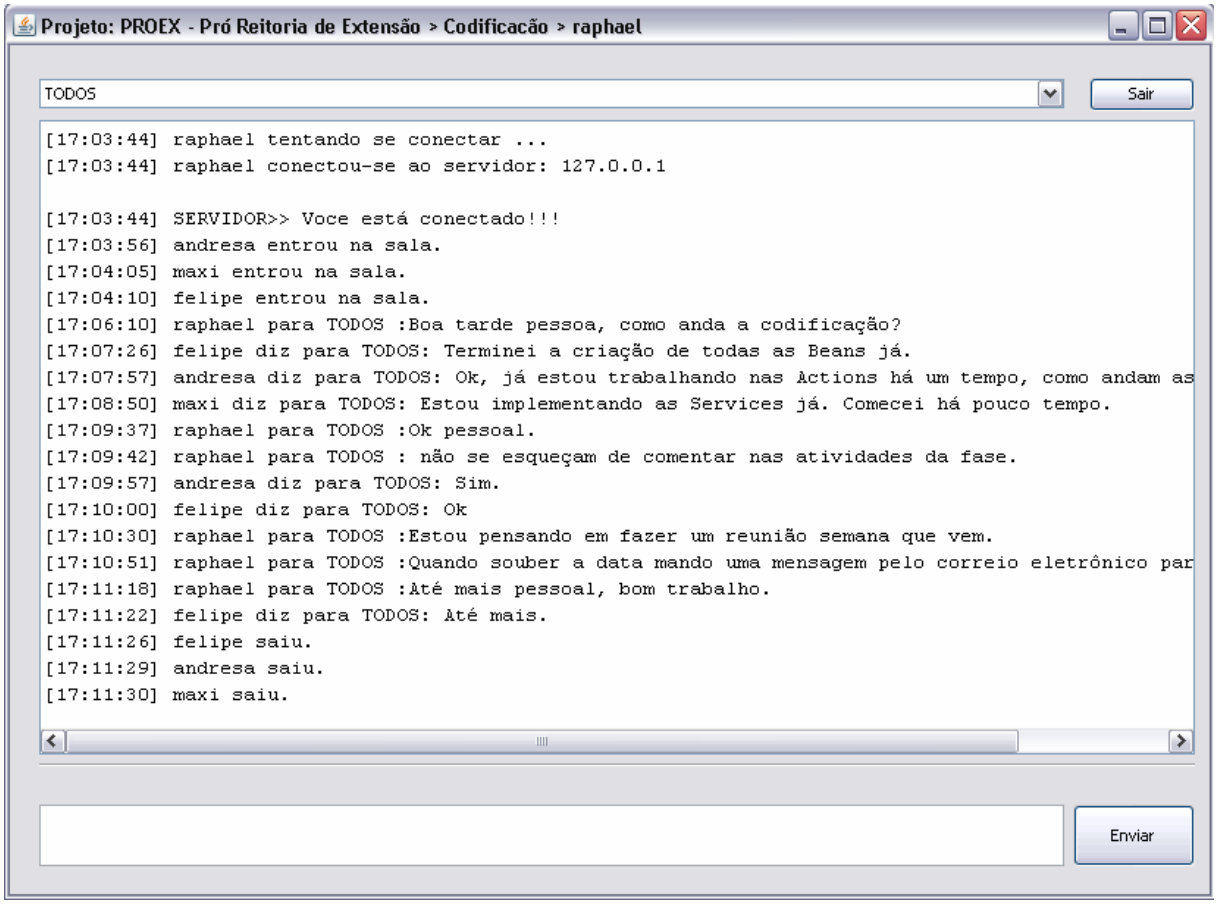

Figura 41: Interface Chat.

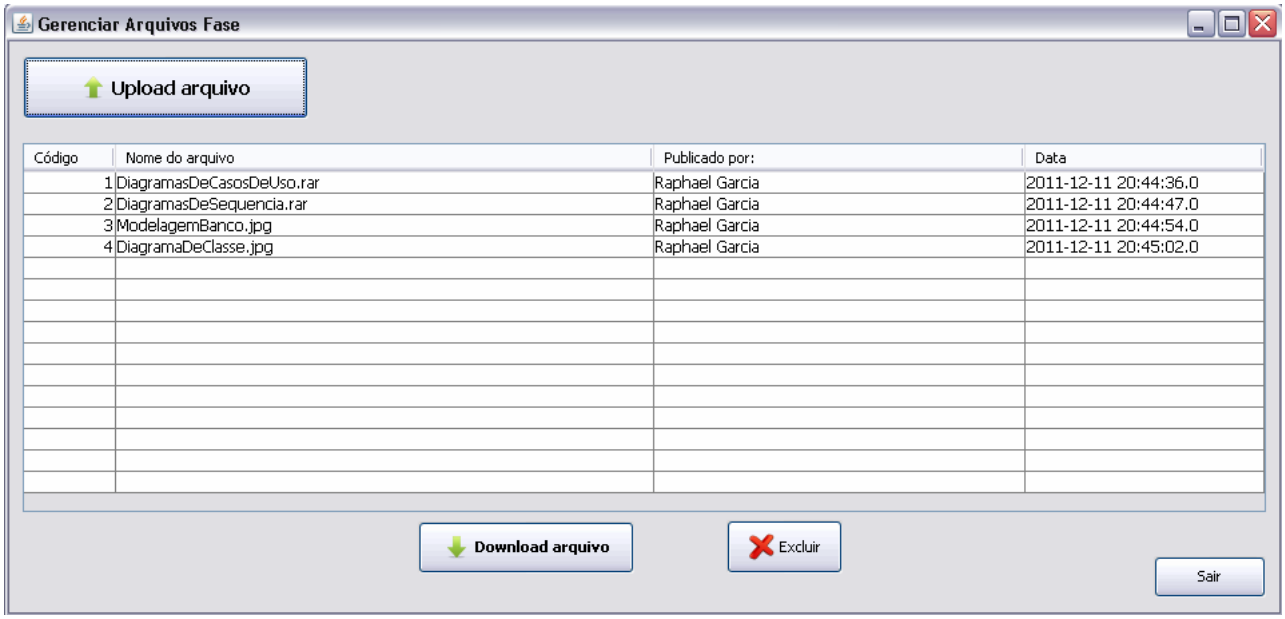

Figura 42: Interface Gerenciar Repositório de Documentos da Fase.

interface de gerenciamento é bastante parecida com a do Chat, como é ilustrado na Figura 43.

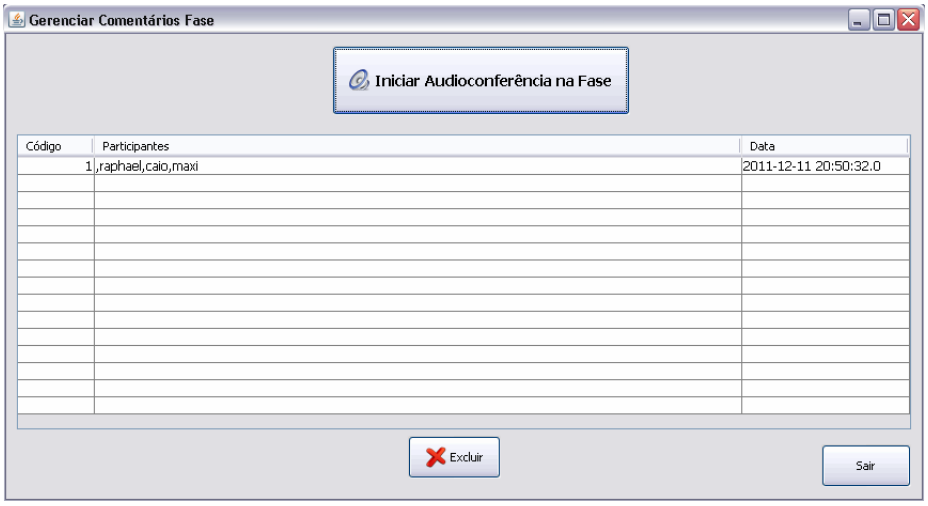

Figura 43: Interface Gerenciar Audioconferências da Fase.

Quando o usuário inicia uma nova audioconferência, ele passa a trasmitir e receber o áudio de todos os usuários que estiverem transmitindo naquela mesma fase ou atividade. Uma lista com o nome dos usuários online é atualizada sempre que um usuário entra ou sai, como ilustra a Figura 44 a qual contém uma audioconferência em andamento entre dois usuários.

Por fim, na interface principal do projeto contém uma ferramenta, também colabora-

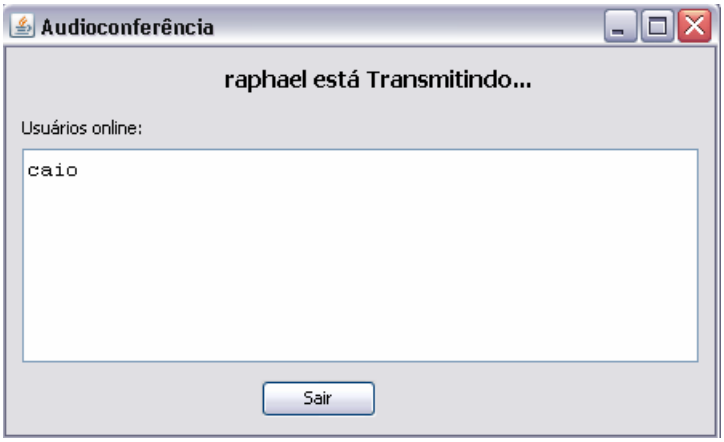

Figura 44: Interface Transmissão de Audioconferência.

tiva, que funciona como uma espécie de correio eletrônico. Cada usuário possui uma caixa de mensagens onde ele pode enviar mensagens para um ou mais usuários do sistema. A Figura 45 ilustra a interface da caixa de mensagem de um usuário.

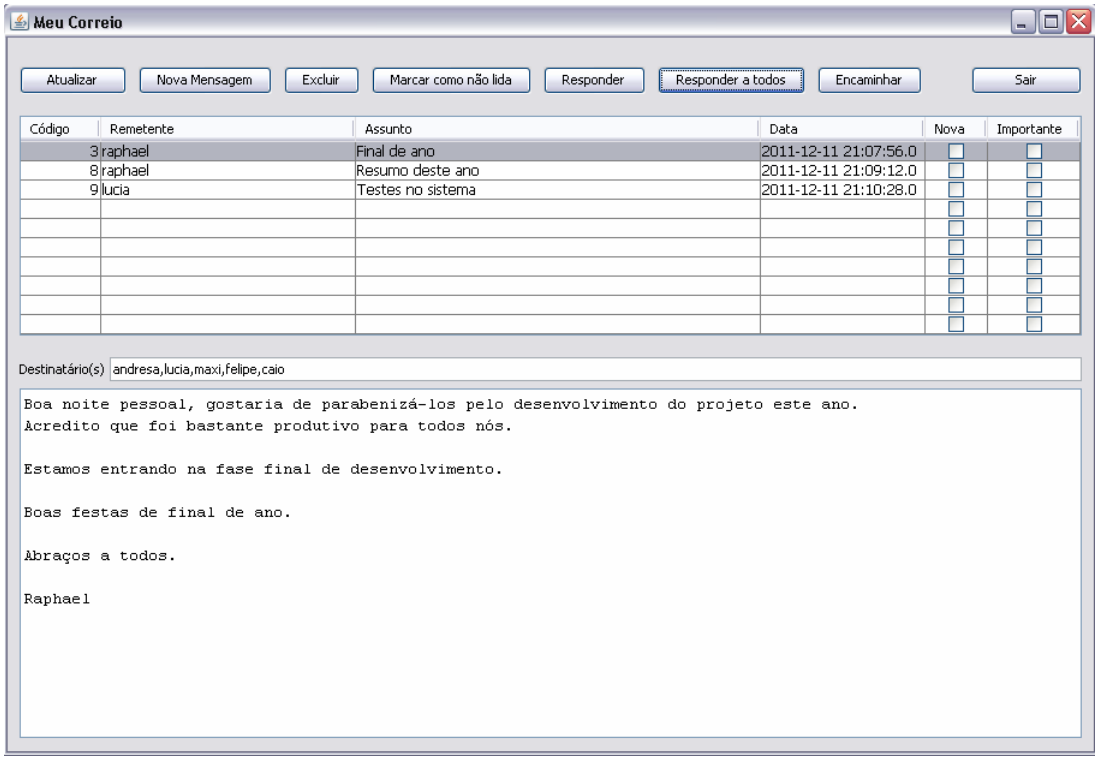

Figura 45: Interface Meu Correio.

Na Figura 45, pode-se observar as mensagens recebidas, juntamente com um painel de visualização abaixo. Existem alguma opções as quais ajudam na manipulação das mensagens, sendo possível encaminhar ou respondê-las de forma prática. Este correio eletrônico ajuda um usuário a enviar uma mensagem a todos os envovolvidos em um projeto por exemplo. Assim que os usuários se autenticam no sistema, o ícone da opção

Meu Correio indica que há novas mensagens, tornando o processo mais prático.

Para avaliar a opinião dos usuários do sistema, foi apresentado o seguinte questionário para quatro dos seis desenvolvedores do sistema Proex:

1) Ter uma ferramenta para gerenciar projetos contando com ferramentas colaborativas na sua opinião é:

[ ] Bom [ ] Indiferente [ ] Ruim

2) Você considera que a interface do sistema é intuitiva?

[ ] Sim [ ] Não

3) Qual a ferramenta colaborativa pela qual você mais se interessou?

[ ] Bate-papo [ ] Comentários [ ] Repositório de Documentos [ ] Audioconferência

[ ] Videoconferência [ ] Correio Eletrônico

4) Qual(is) o(s) recurso(s) que falta(m) no sistema?

Através das respostas obtidas, 100% dos questionados opinaram que é bom ter uma ferramenta que gerencie projetos colaborativamente e também consideraram a interface intuitiva.

A ferramenta colaborativa pela qual houve maior interesse foi o Correio Eletrônico, onde 100% dos questionadas optaram por ele. Um dos usuários assinalou todas as ferramentas colaborativos, justificando que o fato de a PGCP oferecê-las a diferencia de grande parte das ferramentas de gerência de projetos disponíveis no mercado.

Para os questionados, os recursos que faltam são:

- Gráfico de Gantt com a possibilidade de aumentar ou diminuir o tamanho.
- Relátórios gerais do projeto.
- Controle de versão dos arquivos do repositório.

Dois usuários responderam que não faltam recursos, onde um deles analisou que apesar de a PGCP ser intuiva, a interface poderia ser melhorada.

Portanto, a aceitação por parte dos usuários foi considerada satisfatória, mostrando que é possível a utilização da ferramenta em um ambiente empresarial.

Capítulo 5

Conclusão

Por fim, o objetivo de integrar uma ferramenta de gerenciamento de projeto com ferramentas colaborativas foi alcançado. Os resultados obtidos foram satisfatórios e condizem com o escopo do sistema.

Através do estudo de caso foi possível verificar que a PGCP, apesar de não possuir alguns recursos relatados, foi bem aceita pelos usuários que a utilizaram. Desta forma, pode-se concluir que um software que ofereça recursos de gerência de projetos juntamente com ferramentas de colaboração, pode ajudar no estreitamento das barreiras de espaço e tempo entre integrantes de projetos.

Pode-se observar também que apesar de a PGCP oferecer um considerável aumento na comunicatividade, algumas regras de permissões são respeitadas, onde apenas usuários detinados à uma mesma tarefa podem incluir informações e se comunicarem. Porém, para que usuários destinados a tarefas distintas possam trocar informações, há a opção do correio eletrônico, o que não altera o projeto e suas permissões. Além disso, como qualquer inserção de dados na PGCP é identificada, usuários não podem publicar informações anonimamente e nem em nome de outrem. Tais fatores são de suma importância para que uma gerência de projeto possa ser bem realizada, evitando a inclusão de dados falsos, ou a "poluição"do projeto com informações trocadas entre usuários indevidamente.

Uma das ferramentas colaborativas implementadas na PGCP, o chat, está sendo usada como meio de comunicação no Serviço Técnico de Informática da Unesp de Presidente Prudente, do qual o autor desta monografia faz parte do quadro de funcionários no momento. Há ainda um estudo sendo elaborado para que a ferramenta seja implantada em toda Unesp de Presidente Prudente. Apesar de a ferramenta ter sido usado para outro

propósito que não o suporte a gerência de projetos, fica claro um aumento na comunicação por meio de seu uso.

O desenvolvimento deste trabalho de conclusão de curso contribuiu para fixar e utilizar os conhecimentos obtidos no decorrer do curso, além de permitir o conhecimento de tecnologias e frameworks até então desconhecidos, contribuindo para um aumento no nível profissional.

Referências

[1] PRESSMAN, S. R. *Engenharia de software*. Rio de Janeiro: McGraw Hill, 2002.

[2] OLIVEIRA, C. Sistemas colaborativos: Conceitos, características e funcionalidades, 09 2006. Disponível em: http://imasters.com.br/artigo/4655/gerencia . Acesso em: 28/11/2011.

[3] OLIVEIRA, C. Tecnologia da informação e comunicação, 07 2006. Disponível em: http://imasters.com.br/artigo/4412/tecnologia/ . Acesso em: 28/11/2011.

[4] OLIVER, P. Trabalho colaborativo - o nome do jogo é envolvimento e compromisso, 06 2011. Disponível em: http://paulooliver-seller.blogspot.com/2011/06/trabalho-colaborativo-o-nome-do-jogo-e.html . Acesso em: 28/11/2011.

[5] OLIVEIRA, C. Sistemas colaborativos, 08 2006. Disponível em:http://imasters.com.br/artigo/4482/gerencia/ . Acesso em: 28/11/2011.

[6] CAMARGO, L. A. B. D. Gestão colaborativa. *Metrô de São Paulo. São Paulo - SP. Dias 26/11/2004, 02 e 03/12/2004.*, 2004.

[7] COSTA, G. D. *Java em rede - recursos avançados de programação*. Brasport, 2008.

[8] ORACLE. Netbeans, Disponível em: http://netbeans.org/ . Acesso em 01/12/2011.

[9] SUEHRING, S. *Mysql a bíblia*. Editora Campos, 2002.

[10] FOROUZAN, B. A. *Comunicação de dados e redes de computadores*. Bookman, 2006.

[11] RICHARD HULT, M. H. Planner - ferramenta de gerenciamento de projeto do gnome, Disponível em: http://live.gnome.org/Planner/ . Acesso em 30/11/2012.

[12] SOFTWARE, S. Openproj - gerenciamento de projeto, Disponível em: http://sourceforge.net/projects/openproj/ . Acesso em 01/12/2011.

[13] MICROSOFT. Microsoft project - gerenciamento de projeto., Disponível em: http://www.microsoft.com/project/en-us/project-management.aspx . Acesso em: 01/12/2011.

[14] 37SIGNALS. Basecamp, brings project teams together., Disponível em:http://basecamphq.com/ . Acesso em: 01/12/2011.

APÊNDICE A – Documento de Requisitos

Universidade Estadual Paulista "Julio de Mesquita Filho"
Unesp

Especificação dos Requisitos do Software:

Plataforma para Gerenciamento Colaborativo de Projetos

Responsável pelo documento:

Leandro Meira Marinho Queiroz

Presidente Prudente

Especificação dos Requisitos do Software “Plataforma para Gerenciamento Colaborativo de Projetos”

Sumário

Especificação dos Requisitos do Software “Plataforma para Gerenciamento Colaborativo de Projetos” 2
Sumário....2
1 Introdução....3
2 Descrição geral do produto....4
3 Requisitos específicos....5
4 Apêndices....8

1 Introdução

1.1 Propósito do Documento de Requisitos

Este documento tem por objetivo descrever os requisitos de uma plataforma para gerenciamento colaborativo de projetos. Seus leitores (usuários deste documento) são pessoas que tenham interesse em usar o sistema e o autor deste projeto.

1.2 Escopo do produto

1.2.1 Nome do produto

Plataforma para Gerenciamento Colaborativo de Projetos.

1.2.2 Missão do produto

O sistema a ser desenvolvido tem por objetivo ajudar empresas e grupos de pessoas no gerenciamento e desenvolvimento de projetos que independam da posição geográfica de cada pessoa. Este software contará com uma interface baseada em fases de projeto. O usuário poderá utilizar recursos interativos isoladamente para cada fase. Haverá um servidor no qual toda a informação deverá ser mantida salva.

1.2.3 Limites do produto

O sistema não possuirá funcionalidades que possibilitem a gravação das conversas de áudio e vídeo.

1.2.4 Benefícios do produto

Benefício 1: Facilitar a comunicação entre usuários dos sistema.

Benefício 2: Facilitar o gerenciamento de projetos por parte da gerência.

Benefício 3: Facilitar o compartilhamento de dados e informações.

Benefício 4: Facilitar o acompanhamento dos projetos por parte de todos, possibilitando a visualização das fases já completadas.

1.3 Definições, Acrônimos e Abreviações.

Número de ordem	Sigla	Definição
1	HD	Disco Rígido
2	RAM	Memória RAM
3	MB	Megabytes
4	CD	Compact Disc
5	DVD	Digital Video Disc
6	TAPE	Fita demo tape

1.4 Referências

Não aplicável.

1.5 Visão geral do restante do documento

De acordo com o Padrão para Especificação de Requisitos de Software, ou seja:
Parte 2: Descrição geral do produto
Parte 3: Requisitos específicos
Parte 4: Apêndices - Informação de suporte.

2 Descrição geral do produto

2.1 Perspectiva do produto

Espera-se que com o uso do sistema haja uma facilitação na organização e execução de projetos.

2.1.1 Interfaces de hardware

Para que o sistema funcione, deverá haver um servidor conectado à internet e rodando o módulo de servidor do software.

2.1.2 Interfaces de software

Os dados mantidos pelo sistema deverão ser armazenados em um banco de dados utilizando um sistema gerenciador de banco de dados.

2.1.3 Interfaces do Usuário

Esta seção foi deixada a cargo dos Programadores.

2.1.4 Interfaces de comunicação

Na utilização dos recursos de áudio e videoconferência, deverá ser usado microfone, webcam e alto-falantes.

2.1.5 Restrições de memória

Número de ordem	Tipo de memória	Limites aplicáveis
1	HD	O produto deve ocupar no máximo 50 MB (Sem considerar as bases de dados que estarão localizadas apenas no servidor).
2	RAM	O produto deve executar em computadores com 512 MB ou mais.

2.1.6 Modos de operação

O sistema deverá operar em dois modos distintos:

- Modo servidor. Um computador deverá rodar este modo para que seja possível a conexão dos usuários à base de dados e outras informações.

- Modo usuário. Cada usuário deverá rodar este modo em seu computador, que por sua vez se conectará ao modo servidor, permitindo o acesso aos dados e a interatividade com os outros usuários.

2.1.7 Requisitos de adaptação ao ambiente

Número de ordem	Requisito	Detalhes
1	Configurações de microfone, webcam e alto-falantes.	As configurações de microfone, webcam e alto-falantes deverão ser adaptadas de modo a ser compatíveis com o sistema operacional utilizado.

2.2 Funcionalidades do produto

O sistema deverá manter os dados de projetos, como fases, arquivos, dados dos usuários, e permitir ao usuário visualizá-los no próprio sistema.

2.3 Características do Usuário

2.3.1 Descrição

Número de ordem	Ator	Definição
1	Usuário	Os usuários do sistema são pessoas interessadas em organizar e otimizar o desenvolvimento de projetos, sendo eles os responsáveis por realizar as operações no sistema.

2.4 Restrições Gerais

Não aplicável.

2.5 Suposições e Dependências

Não aplicável.

3 Requisitos específicos

3.1 Requisitos Funcionais

RF1. O sistema deve possuir dois módulos: servidor e cliente. O módulo servidor será executado apenas uma vez e é o responsável por intermediar as tarefas entre os clientes. O módulo cliente é executado uma vez para cada cliente conectado e apresenta todas as ferramentas do software.

RF1. O sistema deve autenticar cada usuário no início da execução do módulo cliente do sistema. Esta autenticação é feita através do fornecimento de um login e senha do cliente.

RF7. O sistema deve possuir dois modos de operação geral: administrador e usuário. O modo administrador pode criar, alterar e excluir projetos; criar, alterar e excluir usuários do sistema;

além de poder administrar todos os projetos internamente (criação de fases por exemplo). O modo usuário possui privilégios apenas para alterar suas informações pessoais (como senha, por exemplo) e visualizar informações sobre os outros usuários (exceto campos de senha e custo), além de participar de projetos.

RF8. O sistema deve possuir dois modos de operação no projeto: administrador e usuário do projeto. O modo administrador tem acesso total ao projeto, podendo realizar todas as tarefas referente a projetos, além de poder atribuir a um usuário a função de administrador do projeto. O modo usuário tem acesso apenas a visualização das fases e visualização das atividades de fases as quais ele participa. Ele pode excluir ou alterar apenas atividades, comentários e documentos que ele criou, podendo apenas visualizar os criados pelos demais usuários. O usuário também pode iniciar Chats nas fases e nas atividades, desde que este usuário seja um participante da fase.

RF1. O sistema deve possuir uma caixa de correio eletrônico para cada usuário do sistema. Através do correio eletrônico o usuário deverá poder enviar mensagens de texto simples para um ou mais usuários do sistema, podendo especificar o assunto da mensagem, relevância dela (importante ou não), além de poder excluí-la, marcá-la como não lida, respondê-la para um ou todos remetentes e encaminhá-la. O sistema deve avisar, através de um ícone, a existência de novas mensagens a um usuário.

RF1. O sistema deve permitir o gerenciamento dos usuários do sistema, como criação, alteração e exclusão. Estes usuários terão os atributos: nome, login, senha, localização, telefone, e-mail, função, tipo (administrador do sistema ou não) e custo.

RF1. O sistema deve permitir ao usuário a criação, alteração e exclusão de projetos, os quais terão nome, descrição, responsáveis (administradores do projetos), nome da empresa, data de início e data de fim.

RF2. O sistema deve permitir ao usuário a criação de um ou mais projetos, porém o sistema só apresentará um projeto por instância.

RF4. O sistema deve permitir o cadastro de recursos no projeto (usuários). Recursos são usuários do sistema que farão parte do projeto.

RF3. O sistema deve permitir a criação, alteração e exclusão de fases do projeto, bem como a estipulação de prazos para elas serem completadas. As fases deverão possuir as seguintes informações: nome, descrição, data planejada para início, data planejada para término, data executada de início, data executada de término, além de um campo indicando se a fase está finalizada ou não.

RF5. O sistema deve permitir a atribuição de usuários (recursos do projeto) a uma ou mais fases do projeto.

RF6. O sistema deve apresentar um gráfico de barras que representam as fases do projeto, juntamente com uma linha do tempo (Gráfico de Gantt), apresentando as datas planejadas e executadas das fases.

RF1. O sistema deve permitir a criação de uma ou mais atividades, as quais deverão estar contidas dentro de fases. A atividades deverão ter os seguintes atributos: nome, descrição, data de início, data de término, além de um campo indicando se a atividade está finalizada ou não.

RF10. O sistema deve permitir que os usuários realizem comentários sobre as fases e atividades, além de visualizar os comentários feitos pelos outros integrantes do projeto.

RF1. O sistema deve permitir que os usuários realizem bate-papo através de mensagens instantâneas dentro de fases ou atividades.

RF12. O sistema deve permitir que os usuários depositem arquivos em repositórios de documentos, onde cada repositório está acoplado à determinada fase do projeto.

3.2 Requisitos Não Funcionais

3.2.1 Requisitos de desempenho

RNFD1 - Tempo de Abertura do Sistema

O tempo de abertura do sistema não poderá ser maior que 10 segundos (em uma máquina com processador de 1.6 GHz e 1 GB de RAM em funcionamento normal).

RNFD2 - Acesso ao Banco de Dados

O tempo de acesso ao banco de dados não poderá exceder 5 segundos.

3.2.2 Requisitos de Espaço

RNFE1 – Tamanho ocupado pelo sistema

O produto deve ocupar no máximo 50 MB (Sem considerar as bases de dados que estarão armazenadas apenas no servidor).

3.2.3 Requisitos de Facilidade de Uso

RNFFU1 – Interface com o usuário

O sistema deve possuir interface gráfica que seja intuitiva e de fácil aprendizado.

3.2.4 Requisitos de Confiabilidade

RNFC1 – Falhas do sistema

O sistema deverá atender as especificações sem que haja falhas de utilização que possam prejudicar os dados armazenados ou parar (“travar”) o sistema operacional em uso.

RNFC2 – Persistência dos dados

Os dados devem ser persistentes, evitando assim conflitos e problemas.

3.2.5 Requisitos de Portabilidade

RNFP1 – Portabilidade do sistema

O sistema deve ser portátil e independente de plataforma.

3.2.6 Requisitos de Segurança

RNFS1 – Segurança na Integridade dos dados

A cópia de segurança (backup) fica a cargo do administrador do sistema.

3.2.7 Requisitos de Manutenibilidade

RNFS1 – Manutenção do sistema

O sistema deve ser de fácil manutenção.

RNFS1 – Extensão do sistema

A arquitetura do sistema deve ser planejada para facilitar a extensão do mesmo.

RNFS1 – Gerenciamento de Configuração

Os artefatos produzidos durante o desenvolvimento devem ser mantidos sob um Gerenciador de Configuração.

4 Apêndices

Não Aplicável.