ROBERTA MERCIA RODRIGUES DE OLIVEIRA

COLLABPRO

Metodologia para utilização do Modelo 3C de Colaboração nas disciplinas de Engenharia De Software

VITÓRIA DA CONQUISTA – BAHIA
JANEIRO – 2011

UNIVERSIDADE ESTADUAL DO SUDOESTE DA BAHIA
DEPARTAMENTO DE CIÊNCIAS EXATAS

ROBERTA MERCIA RODRIGUES DE OLIVEIRA

COLLABPRO

Metodologia para utilização do Modelo 3C de Colaboração nas disciplinas de Engenharia de Software

Trabalho de Conclusão de Curso apresentado à disciplina Projeto de Computação Supervisionado II do Departamento de Ciências Exatas da UESB, como requisito parcial para a obtenção do título Bacharel em Ciência da Computação.

Orientadora: MsC. Maísa Soares dos Santos Lopes

VITÓRIA DA CONQUISTA – BAHIA
JANEIRO – 2011

ROBERTA MERCIA RODRIGUES DE OLIVEIRA

COLLABPRO

Metodologia para utilização do Modelo 3C de Colaboração nas disciplinas de Engenharia de Software

Trabalho de Conclusão de Curso aprovado como requisito parcial para obtenção do grau de Bacharel no curso de Ciência da Computação da Universidade Estadual do Sudoeste da Bahia.

Banca Examinadora:

Orientador: _____

MsC. Maísa Soares dos Santos Lopes
Universidade Estadual do Sudoeste da Bahia – UESB

Membro 1: _____

Esp. Gidevaldo Novais dos Santos
Faculdade de Tecnologia e Ciências – FTC

Membro 2: _____

Esp. Fabrício de Sousa Pinto
Faculdade de Tecnologia e Ciências – FTC

Vitória da Conquista – BA, 19 de janeiro de 2011

AGRADECIMENTOS

Começo agradecendo a Deus por ter me dado capacidade para ingressar, e permanecer, na graduação e força para vencer os obstáculos e concluir essa jornada.

Aos meus pais pelo amor, carinho, paciência, cuidado e educação que dispensaram a mim.

Aos familiares que acreditaram na minha capacidade.

À amiga Del que nesses cinco anos esteve ao meu lado nas horas boas e ruins.

Aos amigos Bruno, Kely, Mara, Fabiano e Lúcio pela amizade construída e pelos bons momentos vividos.

Ao amigo Fabrício pela amizade e por ter me ajudado com a escolha do tema deste trabalho.

À querida Celina pela amizade, dedicação e por me incentivar a nunca desistir dos meus objetivos.

À querida orientadora Maísa pelo apoio, incentivo e confiança.

E às demais pessoas que de alguma forma contribuíram para a conclusão deste trabalho. A todos, o meu Muito Obrigada!

RESUMO

Com o avanço tecnológico dos últimos anos, a comunicação interpessoal sofreu muitas mudanças. A escassez de tempo para encontros presenciais motivou a utilização de *groupwares* (softwares que possibilitam grupos trabalharem a distância)**.** A colaboração é a chave para que as equipes se relacionem de forma útil e produtiva. Esse trabalho baseia-se no Modelo 3C de Colaboração, segundo o qual a colaboração é composta pela comunicação, coordenação e cooperação. E-mails, fóruns, listas de discussão, mensagens instantâneas e sistemas de vídeo-conferência são alguns exemplos de *groupwares* (também conhecidos como softwares colaborativos). O objetivo desse trabalho foi propor a metodologia CollabPro para otimizar o uso de *groupwares* nas disciplinas da área de Engenharia de Software do curso de Ciência da Computação da Universidade Estadual do Sudoeste da Bahia – UESB. Para isso, além do levantamento bibliográfico acerca dos assuntos inerentes aos sistemas colaborativos, também foi realizada uma pesquisa sobre alguns Ambientes de Desenvolvimento Colaborativo. Para auxiliar no desenvolvimento do trabalho, os alunos do curso foram consultados a respeito da utilização de ferramentas colaborativas durante o desenvolvimento dos trabalhos acadêmicos. A metodologia desenvolvida aplica os conceitos do Modelo 3C e visa auxiliar os trabalhos em equipe através do uso de *groupwares*.

Palavras-chave: *Groupware*, colaboração, Modelo 3C, comunicação, coordenação, cooperação, CollabPro, Ambientes de Desenvolvimento Colaborativo.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 – INTRODUÇÃO

1.1 Contextualização e Motivação

Devido ao grande avanço tecnológico das últimas décadas, o desenvolvimento e a utilização de softwares se popularizaram em diversos segmentos da sociedade. Eles estão presentes na área acadêmica, da saúde, empresarial, científica, entre outras.

Outra consequência desse avanço se dá na área da comunicação. Atualmente é possível se informar de fatos que acontecem no mundo inteiro em tempo real. Com esse fluxo de informações tão intenso, a comunicação interpessoal também sofreu mudanças, a utilização de e-mails e *groupwares* (programas que dão suporte à realização de atividades em grupo) tem crescido bastante, como forma de otimizar a transmissão de mensagens.

No âmbito acadêmico, esses programas auxiliam o desenvolvimento de atividades curriculares, pois são uma alternativa à falta de tempo para reuniões presenciais entre a equipe. A utilização de softwares colaborativos no ambiente acadêmico tem propiciado uma maior interação entre os alunos e, assim, contribuído para dissipar uma gama de informações relativas aos trabalhos do grupo.

O objeto de estudo desse projeto são as disciplinas da área de Engenharia de Software do curso de Ciência da Computação da UESB. Verificar se os alunos utilizam algum tipo de *groupware* para auxiliar na execução de trabalhos acadêmicos e de que forma o utilizam é essencial na elaboração de uma proposta para que os discentes possam utilizar sistemas colaborativos de forma otimizada.

Fazer com que os alunos interajam de forma organizada e permitir um maior contato com o professor durante a realização dessas atividades, são apenas algumas vantagens obtidas após estruturar uma metodologia que permita um uso mais vantajoso dos *groupware*s.

Nessa metodologia são aplicados conceitos do Modelo 3C de Colaboração. Esse modelo é baseado no princípio de que a colaboração é feita a partir de três pilares: comunicação, coordenação e cooperação.

1.2 Objetivos

1.2.1 Objetivo Geral

Desenvolver a CollabPro, uma metodologia de utilização do Modelo 3C de Colaboração nas disciplinas da área de Engenharia de Software, no curso de Ciência da Computação da UESB.

1.2.2 Objetivos Específicos

- Contribuir para a melhoria da comunicação entre integrantes de uma equipe (utilizando ferramentas de *groupware*);
- Permitir maior interação entre docentes e discentes;
- Possibilitar que, mesmo com a mudança dos integrantes de um grupo, os trabalhos realizados em uma disciplina possam ser reutilizados (e melhorados) nas disciplinas subsequentes.

1.3 Metodologia

Para o desenvolvimento deste projeto houve, inicialmente, um estudo sobre engenharia de software e software colaborativo (*groupware*). Posteriormente foi feita uma breve explanação a respeito da importância do trabalho em equipe. Em seguida o modelo 3C de colaboração é apresentado e a partir dele desenvolveu-se a metodologia objeto de estudo deste trabalho.

Convém ressaltar que os estudos desse projeto foram embasados em pesquisas bibliográficas. Também foram utilizadas técnicas de levantamento de informações, para isso houve a aplicação de um questionário em uma amostra dos alunos de Ciência da Computação da UESB.

1.4 Organização da monografia

Nesta subseção foi apresentada uma visão geral do que é abordado nesse trabalho. No capítulo 2 há uma revisão bibliográfica que abrange engenharia de software, a importância do trabalho colaborativo, além de apresentar a definição de *groupware* e da sua engenharia. Nesse capítulo também será apresentado o Modelo 3C de Colaboração.

No capítulo 3 é apresentada uma análise de alguns ambientes de desenvolvimento colaborativo. Já no capítulo 4 é feita uma breve descrição do curso de Ciência da Computação, além da tabulação dos dados colhidos através do questionário, e, finalmente, é proposta a CollabPro. Por fim, no capítulo 5 há a apresentação das conclusões desse trabalho e sugestões para trabalhos futuros.

2 – REVISÃO DE LITERATURA

2.1 Engenharia de Software

Antes de definir a Engenharia de Software, é necessário apresentar uma definição do que vem a ser um software. Há uma tendência em acreditar que software consiste apenas em uma aplicação de computador com uma determinada finalidade. Mas essa definição é mais ampla. Segundo Mendes (2002, p.3) “o termo software não se restringe apenas aos programas de computadores associados a uma aplicação, mas também envolve toda a documentação necessária para instalação, uso, documentação e manutenção dos programas”.

Outra questão importante acerca deste tema, refere-se às suas características. Sommerville (2007, p.5) elencou alguns atributos para caracterizar um software: ele deve atender às expectativas do cliente no que tange ao desempenho e à funcionalidade, deve ser de fácil manutenção, além de ser confiável e usável. É importante que além de usável, ele possa ser reusável. A reusabilidade também é uma característica importante para um software de qualidade. (Pressman, 1995).

Em linhas gerais, pode-se definir Engenharia de Software como a área da Ciência da Computação responsável pelo desenvolvimento e manutenção de sistemas. De acordo com Falbo (2005, p.2) “A Engenharia de Software trata de aspectos relacionados ao estabelecimento de processos, métodos, técnicas, ferramentas e ambientes de suporte ao desenvolvimento de software”.

Para se desenvolver um software com qualidade é necessário haver um planejamento das ações a serem executadas pela equipe de desenvolvimento. Esse conjunto de ações é denominado processo de software.

> Um processo de software é um conjunto de atividades que leva à produção de um produto de software.[...] Os processos de software são complexos e, como todos os processos intelectuais e criativos, dependem de julgamento humano[...] podem ser aprimorados por meio da padronização de processo, na qual a diversidade de processos de software ao longo da organização é reduzida. Isso promove o aprimoramento da comunicação[...] A padronização é também um passo inicial importante na introdução de novos métodos e técnicas de engenharia de software e também nas boas práticas de engenharia de software. (SOMMERVILLE, 2007, p.42-43).

Ainda de acordo com Sommerville (2007, p. 8) “um método de engenharia de software é uma abordagem estruturada para desenvolvimento de software, cujo objetivo é

facilitar a produção de software de alta qualidade dentro de custos adequados".

Apesar de existirem diferentes métodos, pode-se afirmar que todos possuem componentes comuns. Alguns componentes são:

- Descrições de modelos de sistemas – consiste em descrever modelos de sistemas e suas características;
- Regras – restrições que podem ser aplicadas no modelo de sistema em questão;
- Recomendações – princípios que se seguidos garantem uma boa prática de implementação;
- Guia de processo – conjunto e organização das atividades a serem realizadas para o desenvolvimento de modelos de sistemas de forma organizada.

Sommerville (2007) define três tipos de modelo de processo de software:

– **Modelo em cascata** – baseia-se em desenvolver software de forma sequencial, dividindo o processo em várias fases. A Figura 1 ilustra as fases do processo

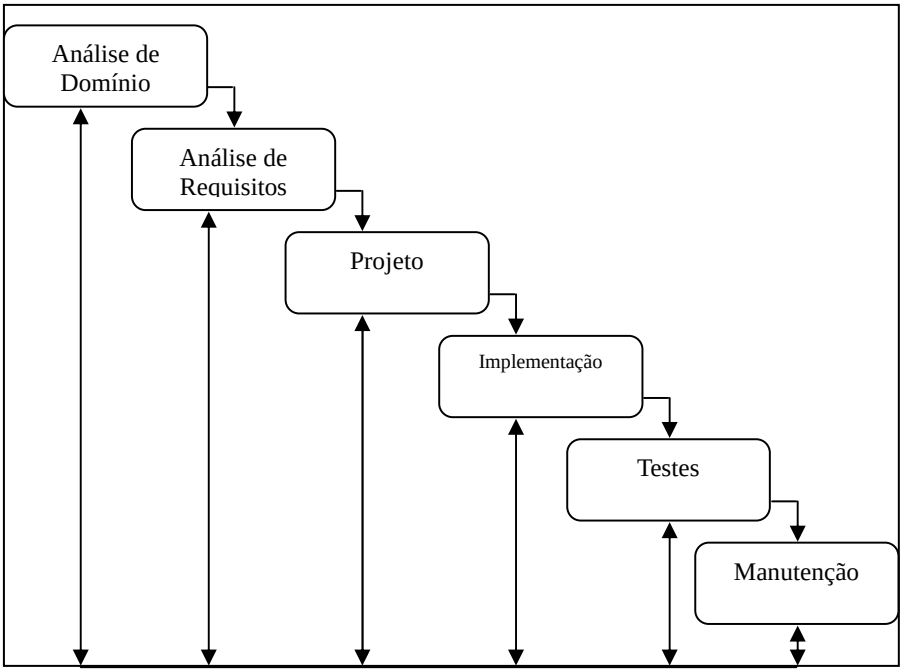


Figura 1: Modelo em cascata (Pressman, 1995).

- Análise de Domínio – é analisado qual será o domínio da aplicação. Arango (1991) define essa etapa como "identificar e organizar o conhecimento sobre uma classe de problemas para suportar a descrição e a

solução destes";

- Análise de Requisitos – fase na qual são identificados os requisitos do sistema, isto é, identificar quais as características necessárias para atender às especificações do cliente;
- Projeto – após definir os requisitos da aplicação, é realizado o projeto do software, analisando o domínio e identificando a melhor linguagem, banco de dados, quais as métricas que serão utilizadas. Um bom projeto de software é crucial para uma boa implementação;
- Implementação – consiste em transformar o projeto em um produto de software. Nessa fase é feita a codificação do sistema;
- Testes – nessa etapa o software é testado a fim de se encontrar erros para posterior correção;
- Manutenção – atualização de informações, incorporação de novos recursos e correção de erros legados são algumas atividades realizadas nessa etapa. A manutenção periódica de um software contribui para aumentar seu tempo de vida.

– **Desenvolvimento evolucionário** – também conhecido como processo iterativo. Nesse processo, ao final de cada etapa (iteração) é obtida uma parte pequena, mas abrangente, do produto final. Isso facilita a identificação de problemas a curto prazo;

– **Desenvolvimento baseado em componentes** – esse modelo prega pela construção de componentes de software (implementação de pequenas funcionalidades) independentes, mas que possam ser interligados para a obtenção do produto final. Essa componentização oferece ao desenvolvedor a possibilidade de reutilizar o software futuramente, o que diminui tempo e esforço.

O processo de engenharia de software pode ser auxiliado por meio de ferramentas. Essas ferramentas são conhecidas como ferramentas CASE *(Computer-Aided Software Engineering).* Pressman (1995, p. 32) afirma que

> As ferramentas de engenharia de software proporcionam apoio automatizado ou semi-automatizado aos métodos[...] [a ferramenta] CASE combina software, hardware e um banco de dados de engenharia de software(uma estrutura e dados contendo importantes informações sobre análise, projeto, codificação e teste) para criar um ambiente de engenharia de software que seja análogo ao projeto auxiliado por computador […] para o hardware.

2.2 *Groupware*

Em linhas gerais, *groupware* (também conhecido como software colaborativo ou cooperativo) pode ser definido como software que dá suporte a atividades realizadas em equipe. Fucks (2002) amplia esse conceito, segundo o autor: "*Groupware* pode ser entendido como a tecnologia baseada em mídia digital que dá suporte às atividades de pessoas organizadas em grupos que podem variar em tamanho, composição e local de trabalho".

Para entender a relevância de um sistema colaborativo, é necessário compreender o que caracteriza um trabalho em grupo e qual a sua importância. Pode-se chamar de trabalho em grupo a reunião ou o agrupamento de indivíduos em prol de uma atividade em comum, ou seja, a equipe trabalha para realizar uma ação que beneficiará a todos. De acordo com Piancastelli (2000) a ideia de equipe é proveniente "da necessidade histórica do homem de somar esforços para alcançar objetivos que, isoladamente, não seriam alcançados ou seriam de forma mais trabalhosa ou inadequada".

Além do benefício após a conclusão da tarefa, o trabalho em grupo também traz motivação para o membro, pois seu trabalho vai ser observado, comentado e avaliado por pessoas de uma comunidade da qual ele faz parte (Benbunan-Fich & Hiltz, 1999).

Desse modo, pode-se claramente entender que com o avanço tecnológico dos últimos anos, é natural que surjam softwares que apóiem o trabalho em equipe, para lidar com certos problemas dos dias atuais, como por exemplo, a falta de tempo para reuniões presenciais. A existência de *groupwares* viabiliza a otimização do tempo para alcançar um objetivo em comum.

Moeckel (2000) acredita que "*groupware* designa a tecnologia (hardware e/ou software) gerada pelas pesquisas sobre CSCW". CSCW pode ser definido como *trabalho cooperativo apoiado por computador.* Esse termo foi construído a partir da evolução da chamada 'automação de escritório'.

2.2.1 Engenharia de *Groupware*

Analogamente ao software comum, o conjunto de técnicas, processos, ferramentas e ambientes que permitem o desenvolvimento de um *groupware* é chamado de Engenharia de *Groupware*. Essa denominação deve-se ao fato de que

> As técnicas e ferramentas da Engenharia de Software que muito evoluíram no desenvolvimento de aplicações monousuário não têm se mostrado suficientes para impulsionar o desenvolvimento de *groupware*. Os sistemas colaborativos usados nas empresas e em grupos de trabalho ainda oferecem um suporte primário para a colaboração entre seus membros. Sistemas colaborativos são complexos de se desenvolver. (GEROSA, 2005)

A Engenharia de *Groupware* deve prover o desenvolvimento de sistemas que possam ser configurados de acordo com as necessidades do grupo de usuários. Essa engenharia possui boa parte dos tópicos abordados na Engenharia de Software, mas devido ao caráter distribuído de um *groupware*, é preciso adequar essa característica durante todo o processo de desenvolvimento desse tipo de aplicativo.

A Figura 2 apresenta um modelo de desenvolvimento em cascata acrescido das especificidades inerentes ao desenvolvimento de um *groupware*

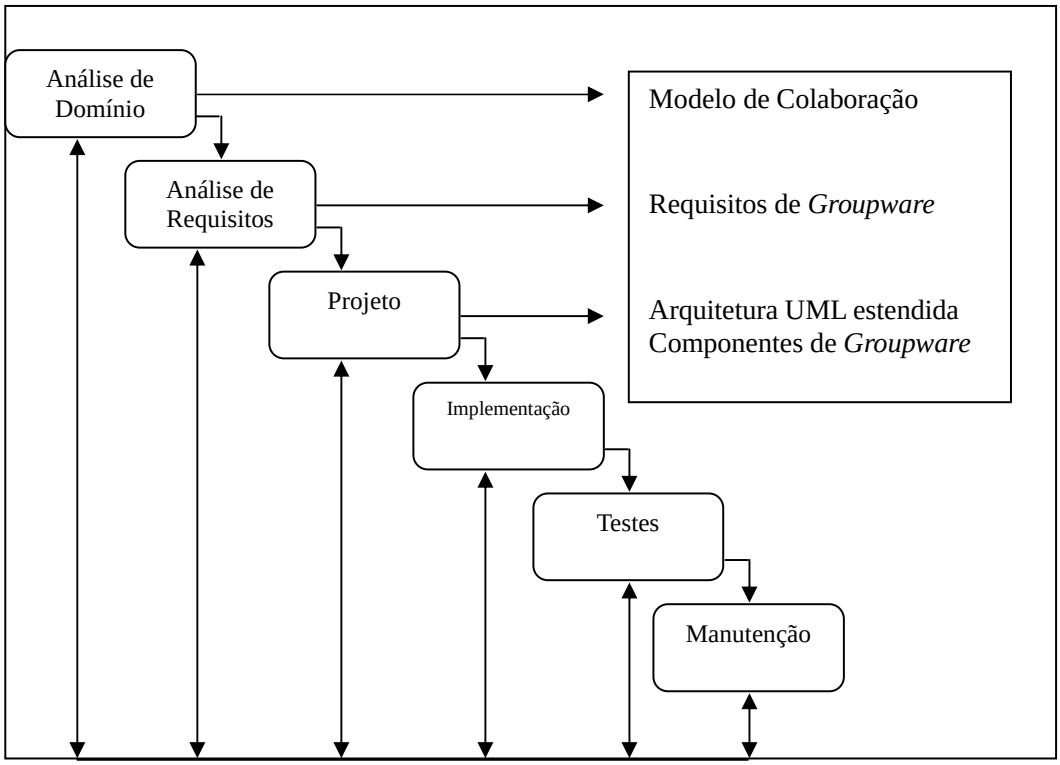


Figura 2: Fases de desenvolvimento de software/*groupware*. (Fucks et. al. 2002)

- Modelo de Colaboração: como já exposto na subseção 2.1, a primeira etapa na elaboração de um software constitui-se da análise de domínio. Para um software colaborativo, é necessário, nessa etapa do processo, escolher um modelo de colaboração para permear todo o desenvolvimento. Um modelo de colaboração consiste na utilização de estratégias que facilitem a comunicação entre membros de uma equipe em busca de um objetivo final comum. Posteriormente será abordado o Modelo 3C de Colaboração, escolhido para embasar

esse trabalho.

- Requisitos de *Groupware*: Devido à complexidade do desenvolvimento de um *groupware*, seus requisitos não podem ser classificados apenas como funcionais, não funcionais. Gerosa (2006) defende que os requisitos de um *groupware* são recorrentes e que apesar de uma lista de requisitos não ser suficiente na especificação completa de um *groupware*, ela auxilia o desenvolvedor a iniciar o processo de criação.

Tietze (2001) propõe um total de 20 requisitos de *groupware*, dos quais onze são classificados como Requisitos do Usuário (RU) e os outros nove são Requisitos do Desenvolvedor (RD). É apresentada, a seguir, uma breve definição de cada requisito.

RU1 – Acesso aos objetos compartilhados e às ferramentas de colaboração – consiste em facilitar o acesso do usuário aos itens (arquivos, ferramentas, etc) colaborativos, garantindo a persistência dos dados e seu compartilhamento;

RU2 – Auxílio na escolha das ferramentas apropriadas – é necessário que o sistema ajude o usuário a escolher a melhor ferramenta para determinada tarefa. Caso haja mais de uma maneira de executa-la, o sistema deve indicar para o usuário a melhor forma de execução;

RU3 – Fornecimento de informações de percepção – o sistema deve emitir alertas referentes à presença de novos usuários conectados à mesma tarefa. Assim todos os usuários terão a percepção de quem está executando determinada tarefa (e como isto é feito);

RU4 – Colaboração síncrona e assíncrona – um *groupware* deve oferecer meios de comunicação entre os usuários, de forma síncrona e assíncrona;

RU5 – Acesso ao ambiente independente da estação de trabalho – prover acesso remoto aos objetos compartilhados;

RU6 – Fornecimento de espaço privativo e público e transição entre eles – apesar de ser colaborativo, é ideal que um *groupware* tenha um espaço restrito para cada usuário, assim cada um terá mais autonomia e privacidade antes de compartilhar algum arquivo;

RU7 – Extensão dinâmica do ambiente – outra característica desejável dos *groupware*s refere-se à sua capacidade de extensão dinâmica, ou seja, agregar novos recursos sem a necessidade de reinicialização do sistema;

RU8 – Sincronização entre ferramentas diferentes – consiste em garantir a sincronia das ações efetuadas sobre um mesmo objeto através de ferramentas distintas;

RU9 – Mobilidade – esse requisito trata da necessidade de adaptar um *groupware* para utilização em dispositivos móveis (realizando as limitações necessárias para tal);

RU10 – Agrupamento de ferramentas – permitir esse agrupamento assegura aos usuários que futuramente o sistema possa adquirir uma configuração já utilizada anteriormente. Isto é

importante para quando ocorrer uma interação similar a outra, com isso a colaboração poderá ser feita de forma mais rápida;

RU11 – Alta performance – uma das características essenciais para um bom *groupware* é que ele seja minimamente afetado pela latência das operações realizadas. É preciso projetá-lo primando sempre pela eficiência. Se o software for demasiadamente lento, o seu uso tende a se tornar inviável;

RD1 – Reuso da experiência e conhecimento anteriores – esse requisito tende a facilitar a integração de novas pessoas à equipe de desenvolvimento e com isso aproveitar ao máximo o conhecimento adquirido na implementação de sistemas mono-usuários;

RD2 – Aproveitamento do modelo de dados – reutilizando modelos de dados já existentes, o tempo de desenvolvimento é diminuído, pois aproveita-se esforços anteriores;

RD3 – Compartilhamento transparente de dados – durante a fase de projeto do *groupware*, os projetistas precisam especificar toda a infra-estrutura para acesso e compartilhamento dos dados, essa não é uma atribuição do programador;

RD4 – Suporte a dados locais e compartilhados – é necessário que a infra-estrutura do software seja desenvolvida de modo a permitir que o desenvolvedor implemente dados locais e compartilhados;

RD5 – Acesso às informações de percepção – o desenvolvedor precisa ter acesso às informações necessárias para oferecer as informações perceptivas aos usuários;

RD6 – Disponibilização de novas ferramentas – as novas ferramentas devem ser disponibilizadas, sem empecilhos arquiteturais, a todos os usuários, afim de que todos utilizem a mesma versão;

RD7 – Escalabilidade – à medida que novos usuários se conectem, a performance do sistema não pode ser perceptivelmente degradada;

RD8 – Integração com ferramentas externas – é interessante que um *groupware* ofereça a possibilidade de integrar novos recursos externos, sendo ou não recursos adquiridos de terceiros;

RD9 – Suporte a ferramentas localizadas no servidor – esse requisito trata o aspecto da confiabilidade do sistema. Alguns recursos do sistema devem ser executados no servidor para melhorar o controle de concorrência do acesso a determinados dados. Mas essa diferença na implementação deve ser imperceptível para o usuário.

- Arquitetura UML estendida – de acordo com Booch et. al. (2000) a *Unified Modeling*

Language (UML) “é uma notação extensível que inclui definições de diagramas de modelagem para as diversas atividades do desenvolvimento, desde as primeiras até as mais refinadas.”

Os diagramas da UML utilizados no desenvolvimento de softwares usuais não contemplam algumas características necessárias para se desenvolver *groupware*. No entanto, a solução para esse impasse deu-se com a utilização desses diagramas com algumas extensões. Daí a nomenclatura UML Estendida.

De acordo com Fucks et.al.(2002), as estruturas acrescentadas aos diagramas são chamadas de estereótipos. Nos diagramas esses estereótipos estão localizados acima do nome da classe, precedido por “<<” e sucedido por “>>”.

Eis algumas extensões utilizadas:

- *Shared* – indica classes que representam objetos compartilhados;
- *Component* – indica classes que representam componentes de *groupware*;
- *Task* – representa uma tarefa;
- *Slot* – utilizado na classe de um objeto compartilhado. Precede o nome dos atributos replicáveis da classe;
- *Session* – refere-se ao espaço onde são realizadas as operações nos objetos compartilhados (sessão).

As Figuras 3 e 4 exemplificam o uso dessas extensões

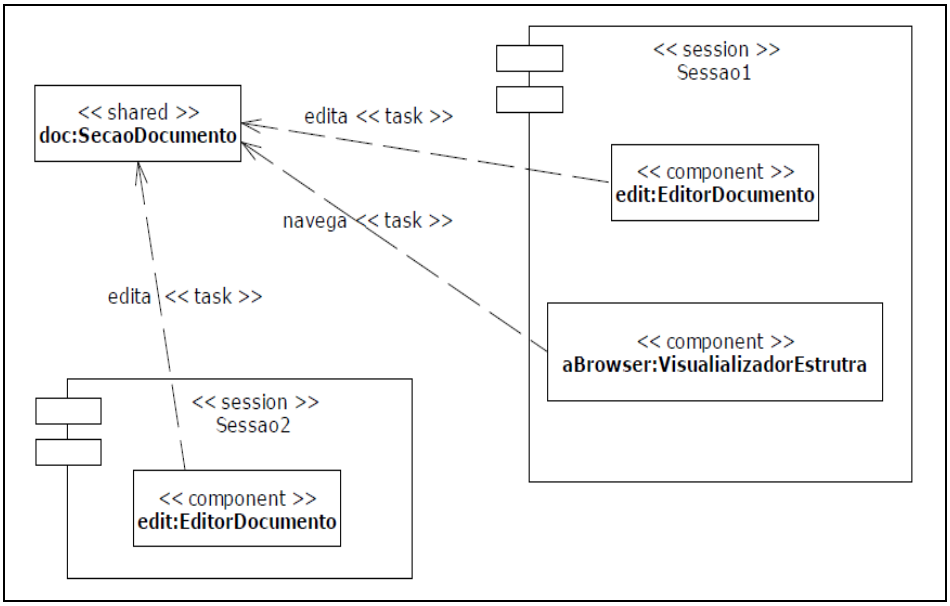


Figura 3: Extensão de UML para modelar sessões (Fucks et.al, 2002)

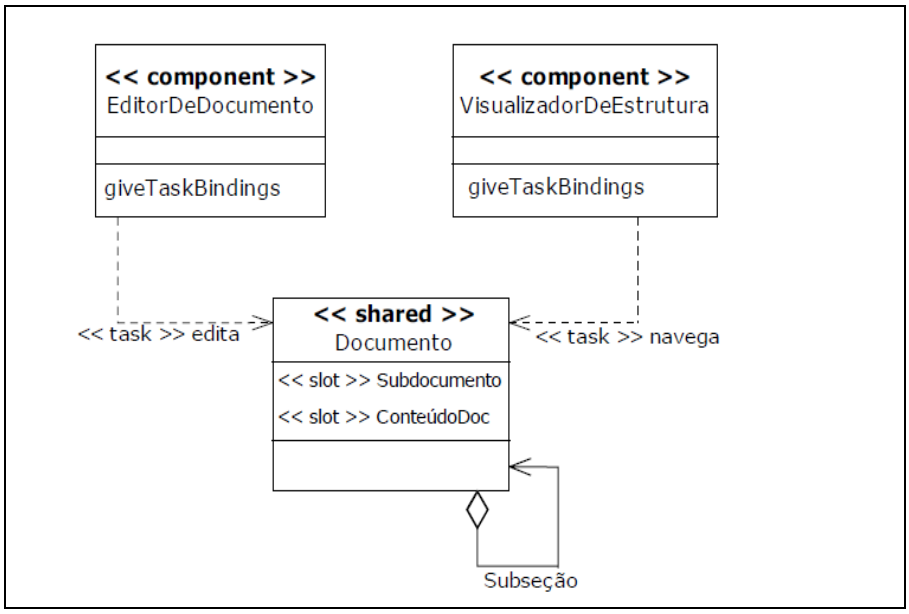


Figura 4: Diagrama de classes apresentando dois componentes de *groupware* que podem executar tarefas em um mesmo objeto compartilhado (Fucks et. Al, 2002).

Na Figura 3 há duas sessões (*Sessao1* e *Sessao2)*. Cada uma possui classes de componentes de *groupware* (*EditorDocumento* e *VisualizadorEstrutura* na *Sessao1* e *EditorDocumento* na *Sessao2*) que realizam as tarefas *edita* e *navega* com os objetos compartilhados da classe *SecaoDocumento*.

Já na Figura 4, há o diagrama de classes da UML estendida, no qual as classes de componentes de *groupware EditorDocumento* e *VisualizadorDeEstrutura* realizam, respectivamente, as tarefas *edita* e *navega* com os objetos compartilhados da classe *Documento.* É possível notar que nesta classe os atributos *Subdocumento* e *Conteúdo* podem ser replicados.

– **Componentes de *Groupware* –** devido à complexidade de implementação de um *groupware*, a componentização é uma alternativa para oferecer flexibilidade a esse tipo de software.

> [As técnicas de desenvolvimento baseado em componentes] visam desenvolver sistemas modulares compostos de componentes de software, que podem ser adaptados e combinados na medida da necessidade, tendo em mente futuras manutenções.[...]Um componente pode fazer parte de outros componentes, e pode ser disponibilizado independentemente, bem como utilizado em composições por terceiros. (FUCKS, 2002).

Geralmente, componentes são desenvolvidos através de *components frameworks.* Johnson (1997) define *framework* como “um projeto reutilizável de todo ou de parte de um sistema, fornecendo um conjunto de classes abstratas e a forma com que suas subclasses

interagem”.

A integração entre as ferramentas em um *component framework* é possível devido à base que este oferece para uma integração dinâmica. Cada ferramenta é vista como um componente de software e os desenvolvedores precisam ter flexibilidade para acoplá-la com outros componentes.

A importância dos componentes não se restringe apenas ao processo de desenvolvimento de um projeto. Os usuários finais precisam ser informados e orientados a respeito da utilização dessas ferramentas (abstraindo detalhes da implementação).

Gerosa (2006) elenca algumas vantagens e dificuldades da componentização de software.

-Vantagens:

- Manutenibilidade – essa característica facilita as alterações nos programas, visto que não será necessário recompilar todo o aplicativo, apenas os componentes que foram atualizados;
- Reuso – diminui o tempo e esforço do desenvolvedor em outros projetos, pois é possível reutilizar coomponentes em diversas aplicações;
- Extensibilidade – com a componentização é possível agregar novas funcionalidades ao software através do acoplamento de novos componentes;
- Encapsulamento – a abstração de detalhes da implementação de um componente facilita o trabalho do desenvolvedor;

-Dificuldades:

- Maior esforço inicial de projeto e implementação – para se desenvolver um componente, é preciso projetá-lo para que seja flexível e escalável. E isso pode demandar grande esforço durante o projeto e a implementação;
- Custo do estudo e entendimento do componente – algumas vezes o custo de se desenvolver um novo componente é inferior ao de estudar um componente já existente para integrar determinada funcionalidade. Por isso a necessidade dos componentes possuírem uma documentação;
- Reuso de componentes de terceiros – dificuldades de acoplamento podem surgir durante a utilização de componentes criados por outro fabricante. Há o risco de surgirem bugs que abalem a confiabilidade do sistema.

Felizmente as vantagens da utilização de componentes superam as dificuldades para a

sua implementação.

2.2.2 Arquiteturas de *Groupware*

Geralmente, as arquiteturas de *groupware* oferecem meios de controlar o compartilhamento, a sincronização de objetos e o controle de concorrência dos mesmos. Nesse trabalho é exposto um tipo de arquitetura de *groupware* baseada em componentes.

> A arquitetura do *groupware* deve prover, portanto, mecanismos de controle de acesso e de concorrência aos objetos compartilhados, para evitar que um usuário sobreponha a ação de outro, pois um mesmo objeto compartilhado pode ser acessado simultaneamente por mais de um componente de *groupware*. É necessária uma forma de mapear os objetos e os componentes associados a eles. (FUCKS et. al., 2002).

Tietze (2001) propõe um modelo de arquitetura baseada em componentes. A Figura 5 representa essa arquitetura.

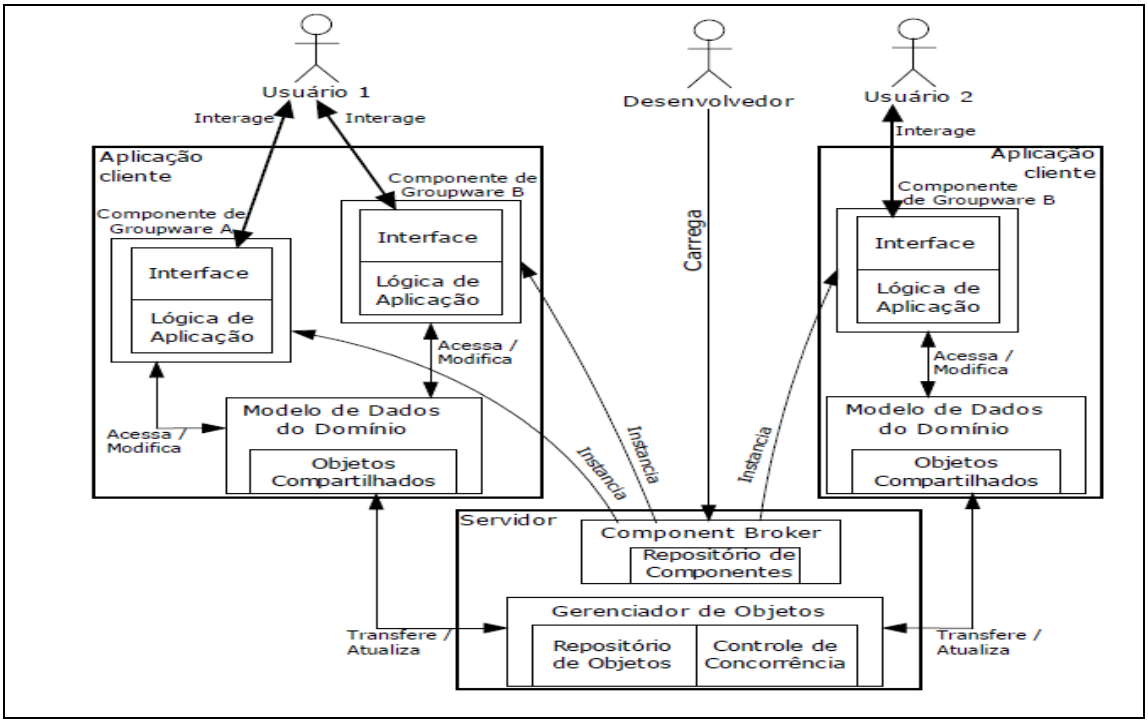


Figura 5: Arquitetura para um *groupware* baseado em componentes (proposta por Tietze, 2001). (Fucks et. al., 2002)

A Figura 5 ilustra a presença de três atores, sendo dois usuários do sistema e um desenvolvedor. O desenvolvedor é responsável pela implementação do sistema e por

alimentar o Repositório de Componentes presente no servidor. Além do repositório, também é encontrado no servidor o Gerenciador de Objetos, composto pelo Repositório de Objetos e pelo Controle de Concorrência.

Os usuários interagem com a Aplicação Cliente, que é composta pelos Componentes de *Groupware* e Modelo de Dados do Domínio. Nesse momento são instanciados os componentes que compõem a aplicação utilizada (através do repositório de componentes). O usuário lida diretamente com a interface dos componentes e, por meio da lógica da aplicação, acessa e modifica o modelo de dados do domínio. No modelo de dados existem réplicas dos objetos compartilhados que foram instanciados para determinada aplicação.

Quando a alteração dos objetos é feita, a aplicação se comunica com o gerenciador de objetos e após verificar qual a última alteração realizada nele (para garantir a integridade e o controle de concorrência dos dados), atualiza a instância da aplicação replicando os objetos alterados para todos os usuários que estão utilizando determinada aplicação.

Deve-se à essa complexidade de arquitetura o fato de um *groupware* bem elaborado necessitar atender a vários requisitos. Desse modo assegura-se a persistência dos dados, controle de concorrência e escalabilidade do sistema.

Existem outras arquiteturas para *groupware*, a seguir será apresentada, brevemente, outra arquitetura que não se baseia na utilização de componentes na sua implementação.

Dewan (1998) propõem uma arquitetura genérica para *groupware* baseada em múltiplas camadas. Cada camada representa um nível de abstração, conforme ilustra a Figura 6:

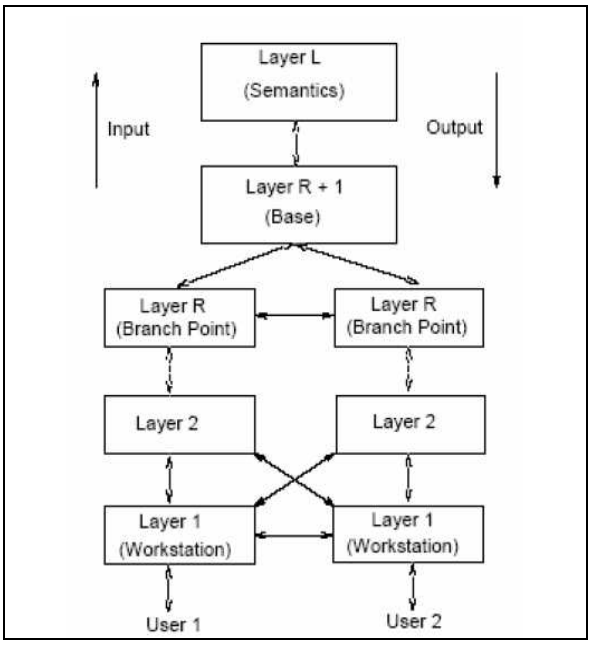


Figura 6: Arquitetura genérica de *groupware* proposta por Dewan (1998) (Gerosa, 2006)

Nesse exemplo há duas ramificações, cada uma utilizada por um usuário. A comunicação entre as camadas é feita através de *feedback* (na mesma ramificação) e *feedthrough* (entre as ramificações).

A camada mais inferior refere-se ao nível do hardware, ou seja, é a camada mais próxima do usuário. Essa camada gerencia os objetos que interagem com a próxima camada e assim sucessivamente, até chegar à camada do nível semântico. Nessa arquitetura, a lógica semântica é trabalhada em um servidor central. Por se tratar de uma arquitetura genérica, não há um número fixo de camadas, ficando a cargo do desenvolvedor decidir quantas camadas serão necessárias para a implementação do seu sistema colaborativo.

2.2.3 Tipos de *Groupware*

Moeckel (2003) classifica os sistemas de apoio à atividades em grupo em duas grandes categorias:

Interação síncrona: sistemas dessa natureza necessitam da presença de todos os usuários envolvidos (independente do local físico em que estejam) num mesmo horário pré-determinado.

Alguns exemplos são:

- Sistemas de Apoio à Tomada de Decisão – Segundo Tiberti(1996), eles “são sistemas interativos, que facilitam a solução de problemas não estruturados por um grupo de pessoas”. O objetivo desses sistemas é aumentar a produtividade e a velocidade com a qual as decisões são tomadas. Sistemas dessa natureza também visam aumentar o aproveitamento de uma reunião;
- Sistemas de Apoio a Reuniões (*meetingware*) – possibilitam que membros de um grupo possam se reunir on-line e viabilizam tanto a comunicação entre eles, quanto o agendamento dos próximos encontros. Geram um registro da reunião de forma automática;
- Editores cooperativos – Como ferramentas de interação síncrona, permitem que vários usuários construam um texto simultaneamente. É possível que um usuário altere o teor do texto, e os outros consigam visualizar essa edição em tempo real;
- Sistemas de Mensagem Instantânea – esses sistemas permitem que os usuários se comuniquem de forma rápida e imediata. Simulando uma conversação em tempo real;
- Sistemas de vídeo-conferência – possibilitam que os usuários interajam utilizando

recursos áudio-visuais. Uma grande vantagem desses programas é possibilitar que inúmeras pessoas acompanhem determinado evento, utilizando recursos financeiros bastante reduzidos.

Interação assíncrona: sistemas desse tipo se caracterizam pela não obrigatoriedade da conexão simultânea dos membros da equipe.

- Workflow – também conhecido como *Gerenciadores de Fluxo de Trabalho,* tem a finalidade de eliminar as ações improdutivas de um processo automatizado, aumentando o fluxo de tarefas;
- Editores cooperativos – já citados na interação síncrona, são sistemas que permitem a edição coletiva de documentos. Sua característica assíncrona permite que os documentos sejam alterados em tempos distintos pelos usuários. As cópias atualizadas desses documentos são automaticamente substituídas a cada nova versão;
- Correio eletrônico (e-mail) – constitui-se na troca de mensagens entre usuários através da Internet. Não necessita de simultaneidade na conexão entre os interlocutores, pois as mensagens são armazenadas nas caixas de entrada e saída do receptor e do emissor, respectivamente;
- Listas de discussão – são utilizadas quando um termo é de interesse comum entre várias pessoas. As mensagens são sinalizadas com o tema em questão e enviada para uma lista de usuários cadastrados, pode ou não haver a presença de um moderador. São largamente utilizadas através de e-mails;
- Fóruns – páginas nas quais usuários leem e comentam sobre um assunto específico. Os fóruns geralmente são organizados em tópicos, para facilitar a localização das mensagens;
- Blogs – são páginas web que contem informações organizadas cronologicamente. O usuário tem a possibilidade de ler e opinar em alguma postagem que considere relevante.

2.3 Modelo 3C de Colaboração

Conforme exposto na subseção 2.2, a relevância do trabalho em grupo motivou a

ideia de desenvolver softwares de apoio às atividades de uma equipe. Além de se alcançar mais rapidamente um objetivo comum, o trabalho em grupo possibilita que cada membro conheça suas dificuldades e promove o crescimento do conhecimento de cada um sobre determinado aspecto.

Gerosa (2006) defende que "para projetar *groupware* de qualidade, é necessário entender de colaboração". Isso significa que não somente os usuários devem colaborar entre si, mas devem utilizar uma ferramenta projetada para esse fim. E, logicamente, para projetar determinada aplicação é preciso conhecer a fundo o seu domínio e isso também se aplica aos sistemas colaborativos.

Ferreira apud Pimentel (2006 p. 18) define que "Colaboração, do latim *co+laborar+ação*, designa a ação de trabalhar em conjunto, a realização de um trabalho em comum realizado por duas ou mais pessoas".

Mas a colaboração não é uma ação única, para colaborar, os indivíduos de uma equipe devem comunicar, coordenar e cooperarem para alcançarem o objetivo proposto. Dessa idéia de colaboração surgiu o Modelo 3C de Colaboração.

De acordo com Gerosa (2006), esse modelo nasceu a partir de um artigo de Elis et. al. (2001) que versava sobre à classificação do suporte computacional à colaboração.Na Figura 7 há a representação do Modelo 3C, destacando a interação entre os 3C's.

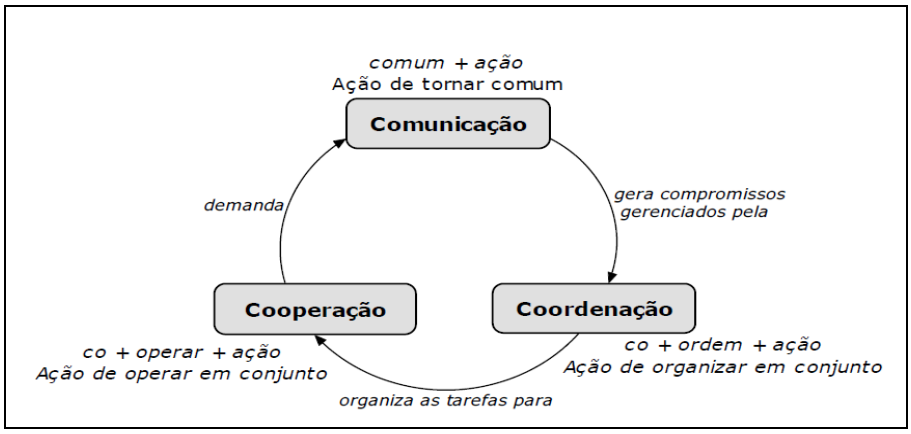


Figura 7: Modelo 3C (Pimentel, 2006)

2.3.1 Comunicação

Um dos princípios básicos da colaboração é a comunicação. Os indivíduos se comunicam para trocar experiências e questionamentos, além de negociar compromissos.

Para uma boa comunicação é necessário que a linguagem utilizada seja entendida por todos os interlocutores. É importante também que as informações não sejam perdidas nem distorcidas. De acordo com Fucks et. al. (2003)

> Os compromissos assumidos nas interações modificam o estado do mundo e têm efeito de ações. Uma ferramenta de comunicação mediada por computador dá suporte às interações entre os participantes, podendo gerenciar as transições de estados, os eventos de diálogo e os compromissos de cada participante.

Para a criação de um software de comunicação, é necessário também seguir esses princípios da boa comunicação. A ferramenta utilizada deve proporcionar uma comunicação adequada entre os usuários.

Fucks et.al (2003) apresenta um modelo de comunicação mediada por computador, conforme é ilustrado na Figura 8.

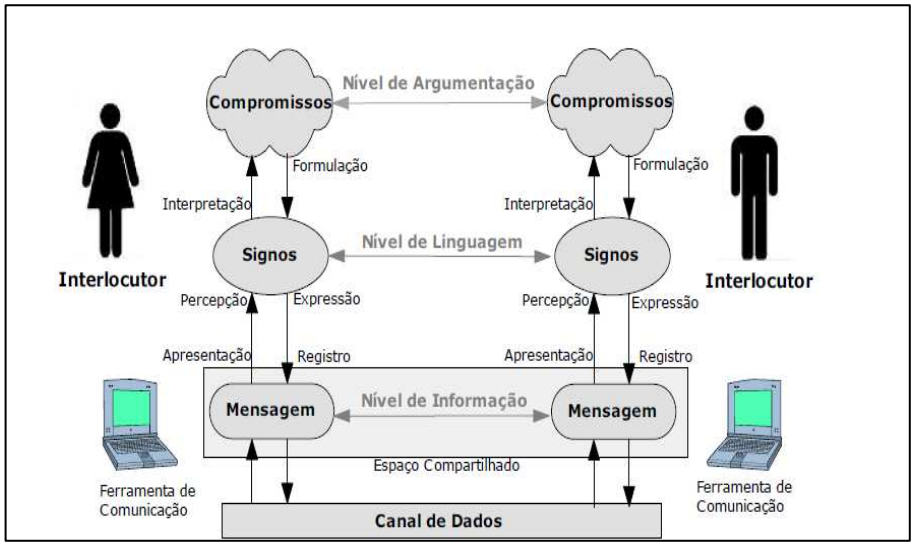


Figura 8: Modelo de comunicação mediada por computador (Fucks et.al. 2003)

Nesse modelo, as informações são transmitidas através do canal de dados (Internet, rede local). Quando o usuário utiliza a ferramenta de comunicação, absorve o conteúdo das mensagens já existentes e efetua a troca de mensagens com os outros participantes (nível de informação).

Convém ressaltar que a associação de signos linguisticos está presente no nível de linguagem , onde pode ser verificado se a linguagem utilizada nas mensagens é compreendida por todos. No nível da argumentação são firmados os compromissos entre os interlocutores.

2.3.2 Coordenação

Durante a comunicação os usuários firmam compromissos e trocam experiências. Mas para que essa comunicação aconteça de forma organizada e produtiva é necessário haver um coordenação nas ações desenvolvidas pelo grupo.

> A coordenação de um trabalho colaborativo objetiva organizar os membros do grupo para que os compromissos resultantes das negociações sejam realizados na ordem e tempo previstos cumprindo seus objetivos e restrições. Também objetiva evitar que esforços de comunicação e de cooperação sejam desperdiçados. (Raposo et. al. *apud* Pimentel(2006))

Segundo Gerosa (2006), "a coordenação de uma atividade envolve a pré-articulação de tarefas, o gerenciamento do seu andamento e a pós-articulação". Pré-articulação consiste no planejamento das ações necessárias para a colaboração. Algumas dessas ações são: identificar os objetivos , mapear esses objetivos em tarefas, selecionar os participantes e distribuir as responsabilidades.

A figura do coordenador é escolhida de acordo com o perfil necessário para exercer essa função. O coordenador deve ter espírito de liderança, conhecimento aprofundado a respeito dos assuntos abordados na colaboração, saber identificar o perfil de cada colaborador e associá-lo à tarefa mais adequada a esse perfil, além de apresentar soluções para resolução de possíveis conflitos entre a equipe. A Figura 9 apresenta uma modelagem de coordenação.

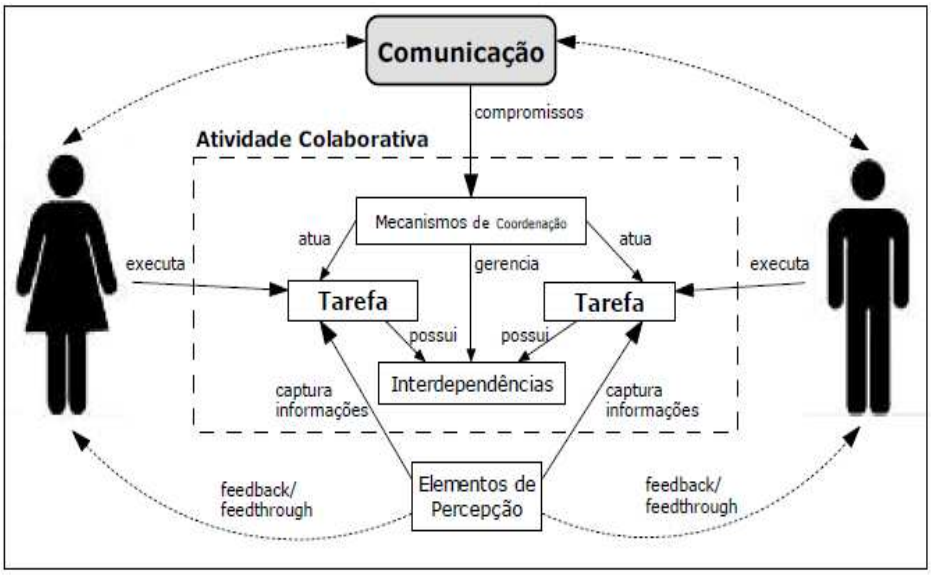


Figura 9: Modelando a coordenação. (Fucks et.al., 2003)

É possível ver na Figura 9 que a coordenação está intrinsecamente ligada à

comunicação. Os mecanismos de colaboração gerenciam as interdependências das tarefas executadas pelos usuários. Nem sempre a figura do coordenador é explícita. De acordo com Fucks et. al. (2003), a coordenação, em algumas ferramentas colaborativas, " fica a cargo do chamado protocolo social, caracterizado pela ausência de mecanismos de coordenação explícitos entre as atividades".

2.3.3 Cooperação

A colaboração necessita da cooperação para que os indivíduos possam trabalhar conjuntamente em prol de um objetivo comum.

> No trabalho em grupo, cooperação é a operação conjunta dos participantes no espaço compartilhado, visando a realização das tarefas. Durante a cooperação, os participantes produzem, manipulam, refinam e organizam objetos, como documentos, planilhas, gráficos, etc. Para atuar nos objetos, os participantes contam com mecanismos de expressão, e para se informar dos resultados de suas atuações (*feedback*) e das ações de seus colegas (*feedthrough*) dispõem de informações de percepção. (GEROSA, 2006).

Os mecanismos de colaboração podem ser síncronos ou assíncronos. Atualmente algumas ferramentas cooperativas oferecem tanto o espaço privativo quanto o compartilhado, possibilitando assim que o usuário possa primeiro desenvolver uma ideia para posteriormente compartilha-la visando uma cooperação.

A cooperação (e a colaboração de modo geral) só é possível devido a elementos de percepção do ambiente. Esses elementos permitem que o usuário tenha uma visão do que foi desenvolvido, quem realizou determinada etapa e quando a mesma foi realizada. Ainda segundo Gerosa (2006) "os elementos da cooperação são relacionados ao registro e recuperação dos objetos e ações".

A Figura 10 ilustra um modelo de cooperação.

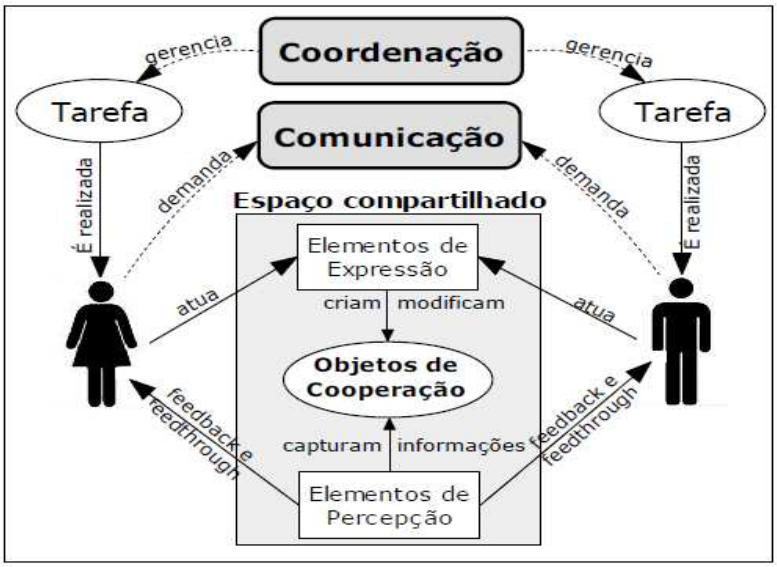


Figura 10: Modelando a cooperação

De acordo com a Figura 10, os objetos da cooperação são criados e modificados pelos elementos de expressão. Eles também captam informações dos elementos de percepção do espaço compartilhado. Esses dois tipos de elementos são constantemente modificados pelos atores que realizam as tarefas definidas pela coordenação.

Após a apresentação dos 3C's desse modelo de colaboração, pode-se perceber que o fundamento básico é a comunicação. Sem comunicação os indivíduos não coordenam nem cooperam. Seria ideal que as ferramentas colaborativas fossem desenvolvidas de modo a atender esses três princípios, porém geralmente as ferramentas tendem a se concentrar em um dos aspectos tratados.

Isso é compreensível, conforme visto nesse trabalho, devido à dificuldade de implementação de um *groupware*. Essa falta de integração entre os três aspectos da colaboração nas ferramentas colaborativas induz os usuários a buscarem várias ferramentas diferentes para interagir com o grupo de trabalho.

A Figura 11 apresenta a classificação de algumas ferramentas de acordo com o Modelo 3C.

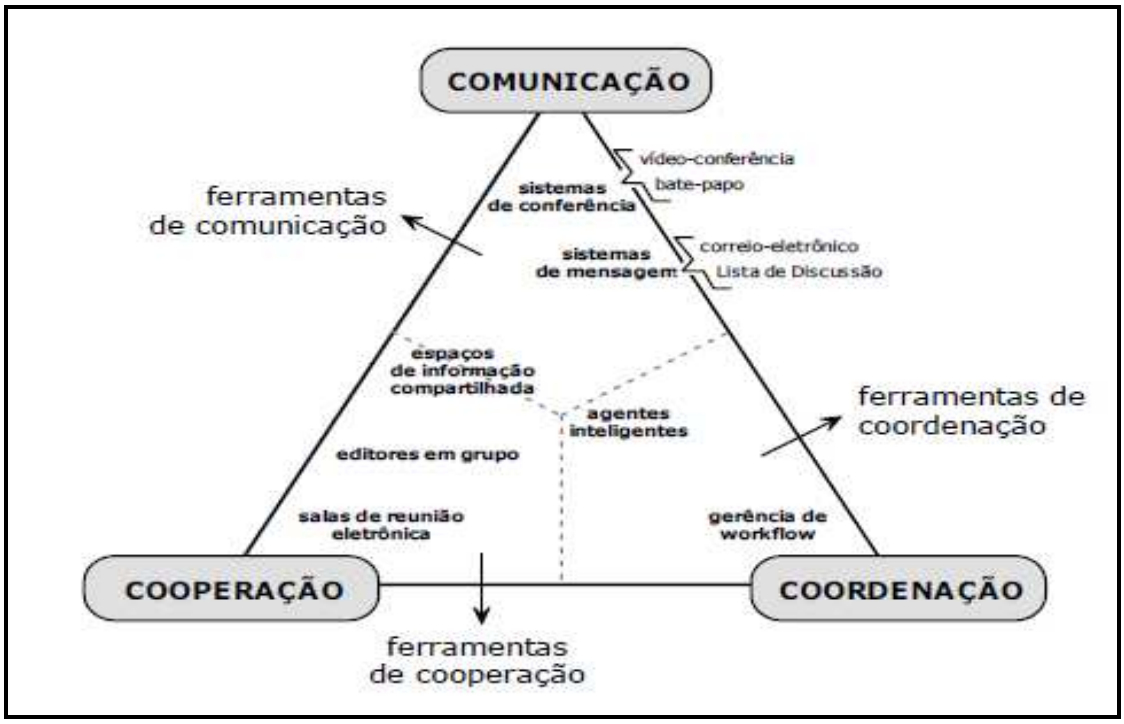


Figura 11: Elementos 3C das ferramentas de comunicação síncrona (Pimentel, 2006)

De acordo com a Figura 11, são apresentadas como ferramentas de comunicação os sistemas de conferência (vídeo-conferência e bate-papo) e os sistemas de mensagens (correio eletrônico e lista de discussão). Já na coordenação estão presentes os sistemas de gerência de workflow e agentes inteligentes. Como ferramentas cooperativas há salas de reunião eletrônica, editores em grupo e espaços de informação compartilhada.

Convém ressaltar que as ferramentas citadas não possuem apenas um aspecto do modelo 3C colaboração. Essa classificação é feita levando em consideração a característica colaborativa mais marcante no sistema.

3 – AMBIENTES DE DESENVOLVIMENTO COLABORATIVO

Ferramentas de colaboração genéricas, como as citadas nesse trabalho, nem sempre atendem às necessidades de uma determinada equipe. Isso pode ser atribuído ao fato de ser necessária a utilização de mais de uma dessas ferramentas para contemplar o desenvolvimento das tarefas do grupo. Nesse caso os usuários precisariam sincronizar a utilização de vários softwares colaborativos para garantir um bom desempenho final.

Como forma de tentar agrupar várias ferramentas colaborativas em uma só, surgiram os chamados Ambientes de Desenvolvimento Colaborativo – ADC (do inglês *Collaborative Development Environment*) são

> motivados pelas necessidades de equipes de desenvolvimento distribuídas, ou virtuais, característicos da globalização do desenvolvimento de software. Entre os requisitos desses ambientes, encontram-se: apoio à encenação de processos em equipes distribuídas, o incentivo à comunicação e à socialização, apoio à atividade individual e em pequenas equipes (2 a 6 pessoas). Tais ambientes objetivam uma maior eficácia e eficiência no desenvolvimento de software e, sobretudo, uma maior satisfação dos participantes (MANGAN, 2006)

Existem, atualmente, inúmeros ADC's, desde os proprietários até os gratuitos. Também existem muitos que são livres (com código-fonte aberto). Esses ambientes são bastante conhecidos por serem largamente utilizados no desenvolvimento de grandes projetos de software livre.

SourceForge[1], *FreshMeat*[2], *Google Code*[3] e *Java.net*[4] são exemplos de ADC's utilizados na construção de grandes projetos com centenas ou milhares de desenvolvedores. Nesse trabalho esses ambientes não serão abordados devido ao foco deste ser o a atividades em pequenas equipes.

Para pequenos projetos, existem ambientes com características similares aos citados anteriormente. Alguns deles são *ProjectPier*, *Trac*, *LibreSource* e *Gforge*.

[1] http://www.sourceforge.net [2] http://freshmeat.net [3] http://code.google.com [4] http://www.java.net

3.1 ProjectPier

ProjectPier (http://www.projectpier.org) é uma comunidade *open source* (código aberto) de desenvolvimento de software. É um gerenciador de projetos baseados na web que utiliza princípios da colaboração em grupo.

Os recursos disponíveis no *ProjectPier* são: mensagens, e-mails de notificação, gerenciamento de tarefas, marcos (*milestones*), gerenciamento de arquivos, tags e formulários.

As mensagens ficam gravadas no site e também são enviadas por e-mail. Essas mensagens podem facilmente ser confundidas com fóruns de discussão, mas segundo os desenvolvedores do ProjectPier, isto se assimila mais à estrutura de um blog.

Atualmente na versão 0.8.6, para instalar esse software, é preciso atender aos seguintes requisitos:

- Servidor Apache 2.0 ou mais recente;
- PHP 5.1 ou mais recente;
- MySQL 4.1 ou mais recente.

Após o programa ser instalado no servidor, os usuários criam contas no sistema e a partir daí podem interagir. Esse software permite um número ilimitado de usuários e, também, permite que um usuário participe de vários projetos. O usuário pode ter acesso aos projetos de qualquer lugar que possua internet.

Figura 12: Página inicial de ProjectPier

Na Figura 12 é apresentada a página inicial do *ProjectPier*. Nela há uma breve introdução sobre o ambiente, a área de login e cadastro de novos usuários, mecanismo de busca e links para: download, manual, tour pelas funcionalidades do sistema, postagens recentes e fóruns ativos.

3.2 Trac

Trac (http://trac.edgewall.org/) é um ambiente colaborativo para gerenciamento de projetos de software. Listas de discussão, interface para Subversion e wiki integrado são seus principais recursos.

A versão mais atual desse software é a 0.12. Para sua instalação, são necessários os seguintes pacotes de software:

- Python versão 2.4 ou superior;

- Setuptool versão 0.6 ou superior (conjunto de melhorias que permitem criar e distribuir pacotes Python mais facilmente);

- Genshi versão 0.6 ou superior (sistema de modelos para renderização de HTML);

- Sistema de banco de dados correspondente com o Python escolhido (SQLite, MySQL ou PostgreSQL)

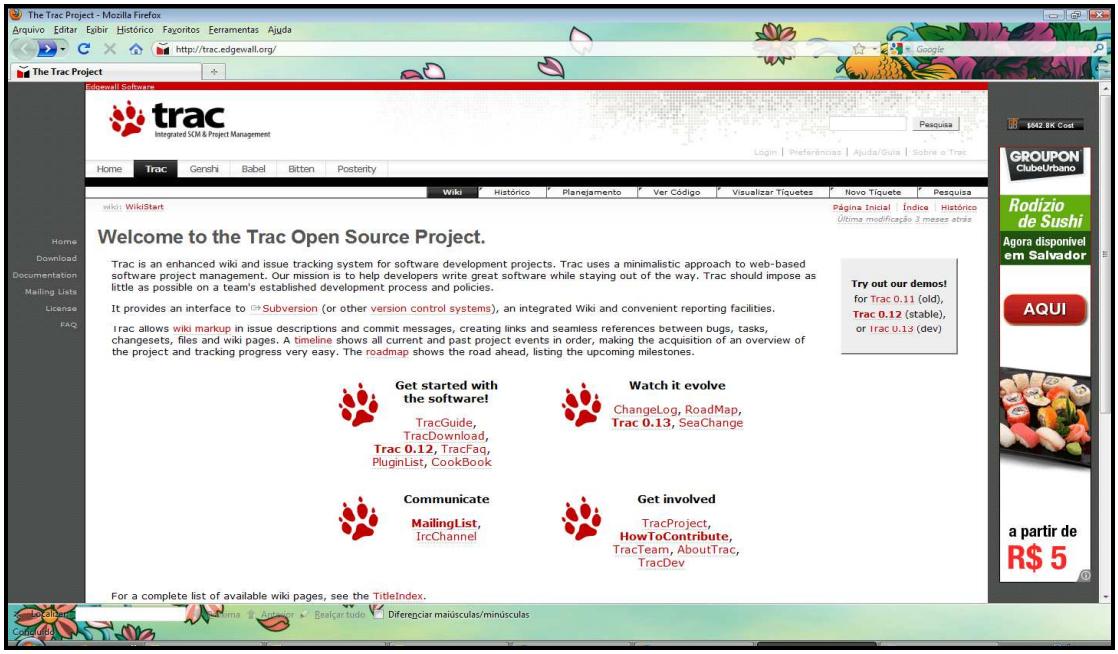


Figura 13: Página inicial Trac

A Figura 13 apresenta a página inicial do *Trac*, contendo, dentre outras coisas, uma introdução sobre o sistema, links para: download, comunicação, perguntas frequentes e campo de busca.

3.3 LibreSource

LibreSource (http://dev.libresource.org) é uma plataforma de colaboração *open-source*, modular e personalizável. É voltada para o desenvolvimento colaborativo de software.

Conta com os seguintes recursos: aplicativo para mensagens instantâneas (Jabber), fóruns, suporte on-line, listas de discussão, edição online de conteúdos, permite integração com o software para controle de versão *Subversion* (http://subversion.tigris.org/).

A versão atual do LibreSource é a 2.5, são requisitos necessários para sua instalação:

- Java JDK 1.5 ou superior;
- PostgreSQL 7.4 ou superior.

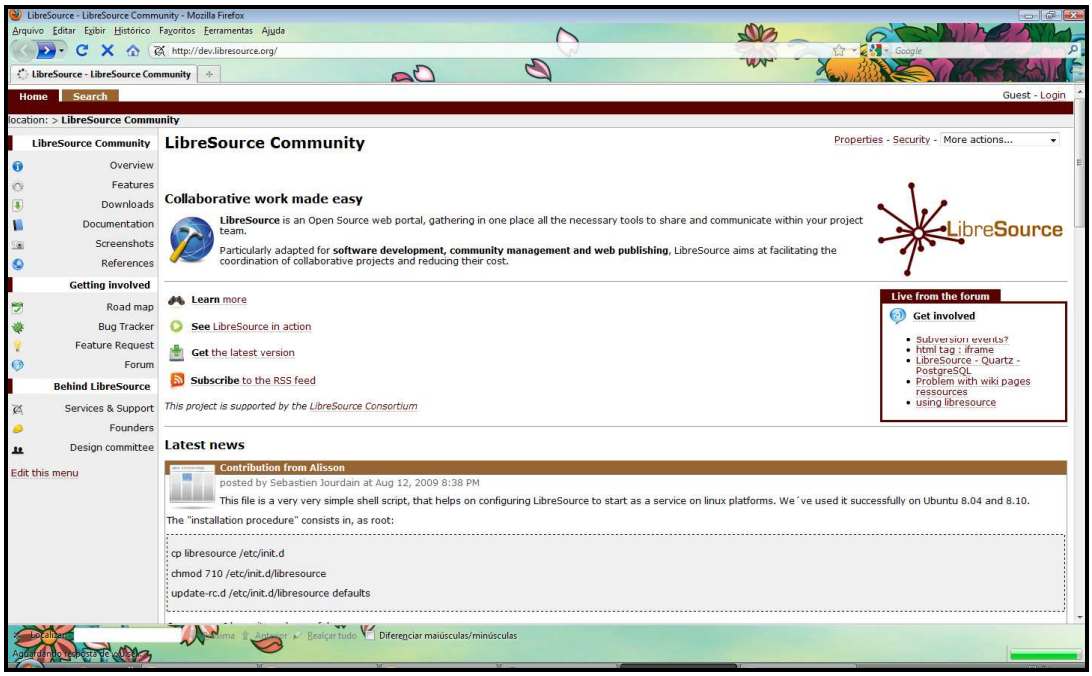


Figura 14: Pagina inicial do LibreSource

Na Figura 14 está ilustrado o conteúdo da página inicial deste ambiente. Estão presentes links para login, *downloads*, documentação, imagens, visão geral, referências. Além disso, há mecanismos de customização do ambiente (a exemplo da opção de editar o menu lateral) e notícias recentes.

3.4 Gforge

Gforge é um ambiente que dispõe de várias ferramentas para auxiliar no desenvolvimento colaborativo de software. Foi criado pelos desenvolvedores do SourceForge como forma de utilizar projetos de menor escala.

A versão *free* (*Community)* que conta com as seguintes ferramentas: fóruns, listas, enquetes, notícias, tarefas, tracker (gerenciador de ocorrências no software), docs (documentação de software), sistemas de controle de versão e arquivos. A versão proprietária e mais robusta, a *Advanced Server* (AS), além de contar com essas ferramentas, é mais expansível e configurável.

Os requisitos necessários para a instalação do Gforge são:

-Sistema Operacional Linux (abrange várias distribuições);

-PostgreSQL 7.1 ou superior;

-Apache 1.3.2 ou superior;

-Openssl 0.9.4 ou posterior;

-PHP 4.0.4 ou superior.

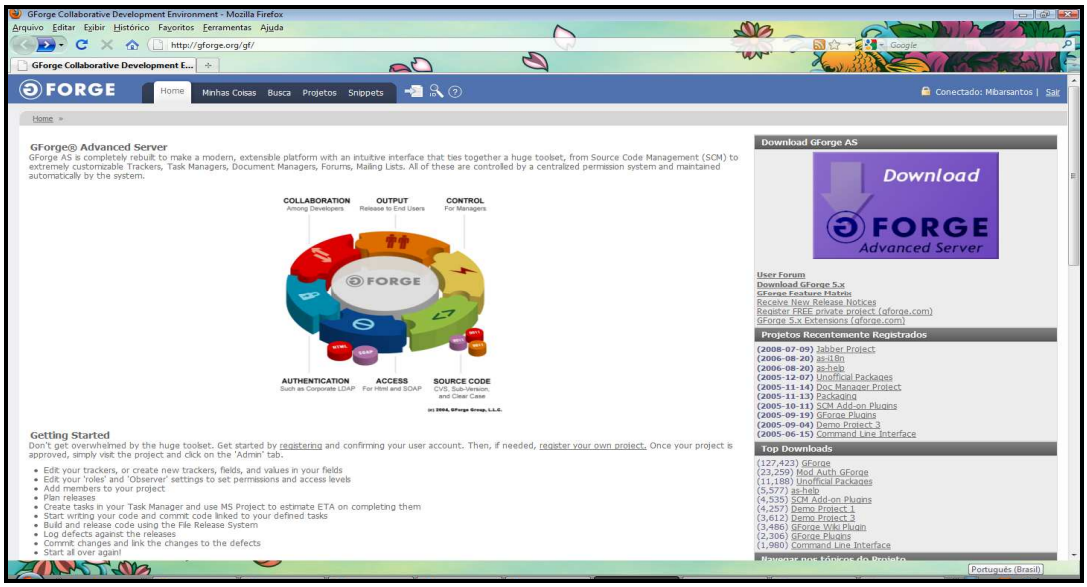


Figura 15: Página inicial Gforge

A página inicial do *Gforge* está ilustrada na Figura 15. Além dos links para login e downloads, há uma breve descrição da versão não gratuita do sistema (*Gforge AS*).

Esse ambiente é bastante utilizado pois permite que os usuários criem e customizem seus próprios projetos. Um exemplo é o site INRIA (http://gforge.inria.fr), serviço que visa facilitar a colaboração de pessoas e herda todas as ferramentas do *Gforge*.

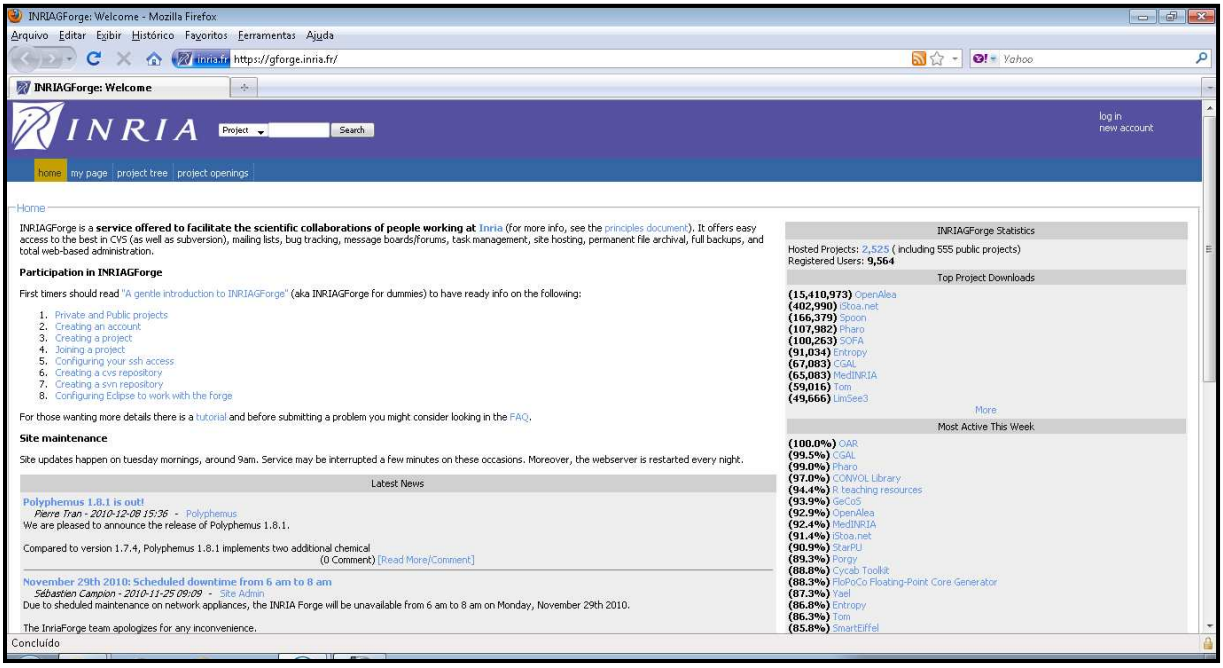


Figura 16: Página inicial INRIA

A Figura 16 ilustra parte do conteúdo do INRIA. Links para login e criação de nova conta, tutorial do ambiente, notícias recentes, estatísticas e busca por projetos são algumas das informações encontradas na página.

4 – DESENVOLVIMENTO DA METODOLOGIA

CollabPro consiste na utilização dos conceitos do Modelo 3C de Colaboração nas disciplinas da área de Engenharia de Software do curso de Ciência da Computação da UESB. Primeiramente é feito um breve resumo da história desse curso, em seguida é apresentada uma análise dos dados coletados no questionário aplicado a uma parte dos alunos do curso (vide apêndice I). Essas são as informações utilizadas no desenvolvimento da metodologia em questão.

4.1 O Curso de Ciência da Computação da UESB

No ano de 1998 foi implantado na UESB o Curso de Bacharelado em Ciência da Computação. Essa implantação só foi possível devido a anos de estudos a respeito da criação de novos cursos de graduação na Instituição e da carência de cursos de Tecnologia da Informação na região sudoeste da Bahia.

O objetivo do curso de Ciência da Computação é

> Formar profissionais com bases científicas e tecnológicas na área de computação, capazes de resolver problemas dos mais diferentes domínios, através de métodos e técnicas computacionais, para atuar de forma bem sucedida tanto na área acadêmica quanto no mercado de trabalho. (PROJETO DE REFORMULAÇÃO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO, 2010).

Segundo informações do Projeto de Reformulação do Curso, nesses 12 anos de existência, o curso formou 154 bacharéis em Ciência da Computação. Em contrapartida, o número de ingressantes nesse mesmo período foi de 512 estudantes. Na Tabela 1 esses dados são apresentados de forma detalhada.

Tabela 1: Ingressos x Egressos. (Projeto de Reformulação do Curso de Ciência da Computação, 2010).

Ano	Número de Ingressos	Número de Egressos
1998	40	---
1999	39	---
2000	40	---
2001	40	---
2002	39	---
2003	40	05
2004	40	07
2005	40	21
2006	38	13
2007	40	39
2008	37	23
2009	39	37
2010	40	09
Total	**512**	**154**

Inicialmente para se tornar bacharel em Ciência da Computação, o aluno precisava concluir o curso em 10 semestres. Mas conforme dados da Tabela 1, a diferença entre alunos que ingressam e os concluintes é muito grande. Alguns alunos abandonam o curso e outros demoram mais tempo que o esperado para sua conclusão.

> O perfil do profissional que se pretende formar no Curso de Ciência da Computação da UESB compõe-se de sólida formação básica, científica e profissional, incluindo aspectos humanísticos, sociais, éticos e ambientais, que o habilite a atuar nas áreas acadêmica, de pesquisa, de gerência de tecnologia de informação, análise de sistemas, gestão, empreendedorismo e consultoria, tanto nas organizações públicas como privadas.(PROJETO DE REFORMULAÇÃO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO, 2010).

Dentro dessa expectativa do perfil do egresso estão contidas quatro características desejáveis para seu bom desempenho profissional. Algumas se destacarão respeitando assim a individualidade do egresso.

- Pesquisador – ter aptidão para trabalhar em pesquisas científicas e/ou tecnológicas;
- Empreendedor – avaliar a viabilidade de determinado projeto, essa característica é altamente vinculada a questões logísticas;
- Consultor – ser apto a prestar consultoria de informática ou computação em diversos tipos de organização;
- Membro de equipe – possuir perfil para coordenar equipes e também ser liderado em projetos da área.

Com base nessas informações, a Coordenação do Curso propôs a Reforma Curricular, diminuindo o tempo da graduação de 10 para 8 semestres (retirando algumas disciplinas das ciências humanas e sociais aplicadas) e dividindo o curso em duas etapas: Ciclo Básico e Perfis. O novo fluxograma do curso está representado no anexo I.

Na primeira etapa todos os alunos cursam as mesmas disciplinas, formando assim uma base de conhecimentos essências para o profissional de computação. A partir do VI semestre inicia-se a segunda etapa, os alunos escolhem em qual área desejam aprofundar seus estudos ou poderão cursar disciplinas optativas de vários perfis.

Os perfis são constituídos de disciplinas optativas e o aluno precisará cursar seis dessas disciplinas para integralizar o curso. Inicialmente está previsto a disponibilidade de quatro perfis: Engenharia de Software e Programação, Sistemas Embarcados, Sistemas Inteligentes e Redes de Computadores & Sistemas Distribuídos.

É importante ressaltar que o independente do perfil escolhido, o aluno, após a formatura, será considerado Bacharel em Ciência da Computação, ou seja, a escolha do perfil não aparecerá no seu diploma de conclusão de curso.

Nesse trabalho o enfoque é dado ao perfil “Engenharia de Software e Programação”. O aluno que optar por esse perfil cursará seis disciplinas das ilustradas na Figura 17.

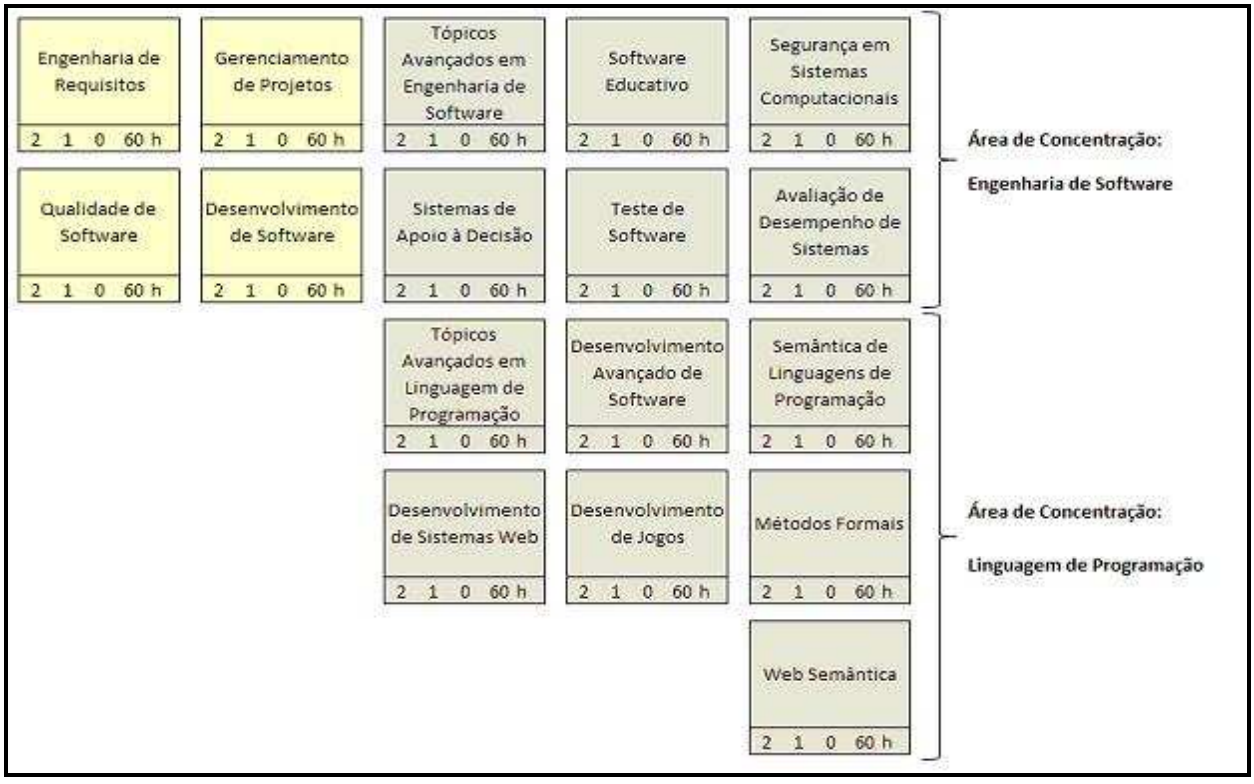


Figura 17: Perfil Engenharia de Software e Programação. (Projeto de Reforma Curricular, 2010)

A metodologia desse estudo foi feita para utilização nas seguintes disciplinas desse perfil: Engenharia de Requisitos, Gerenciamento de Projetos, Qualidade de Software e Desenvolvimento de Software e também em duas matérias do Ciclo Básico são pré-requisitos para as obrigatórias do perfil em questão (Engenharia de Software e Análise e Modelagem de Sistemas). As ementas dessas disciplinas são apresentadas no anexo II.

4.2 Análise dos dados coletados

Atualmente o curso de Ciência da Computação conta com cerca de 200 alunos matriculados. Desses, quarenta e dois responderam ao questionário do apêndice I.

Dos participantes da pesquisa, oito estão matriculados entre o I e o III semestres, nove cursam entre o IV e o VI semestres, seis são alunos do VII ou VII semestres e, por fim, dezenove estão matriculados no IX semestre ou semestres posteriores.

Quando questionados a respeito da importância de realizar trabalhos acadêmicos em grupo, 51% dos estudantes afirmaram que os trabalhos em equipe são importantes, mas todos os membros do grupo devem trabalhar visando um objetivo final único. Trinta e três por cento deles consideram o trabalho em equipe fundamental para o seu desenvolvimento acadêmico. Na opinião de seis em cada cem alunos, a relevância desses trabalhos depende do tipo do mesmo, por fim, apenas 2% dos estudantes não acham tão necessário trabalhar em equipe. O gráfico da Figura 18 ilustra esses dados.

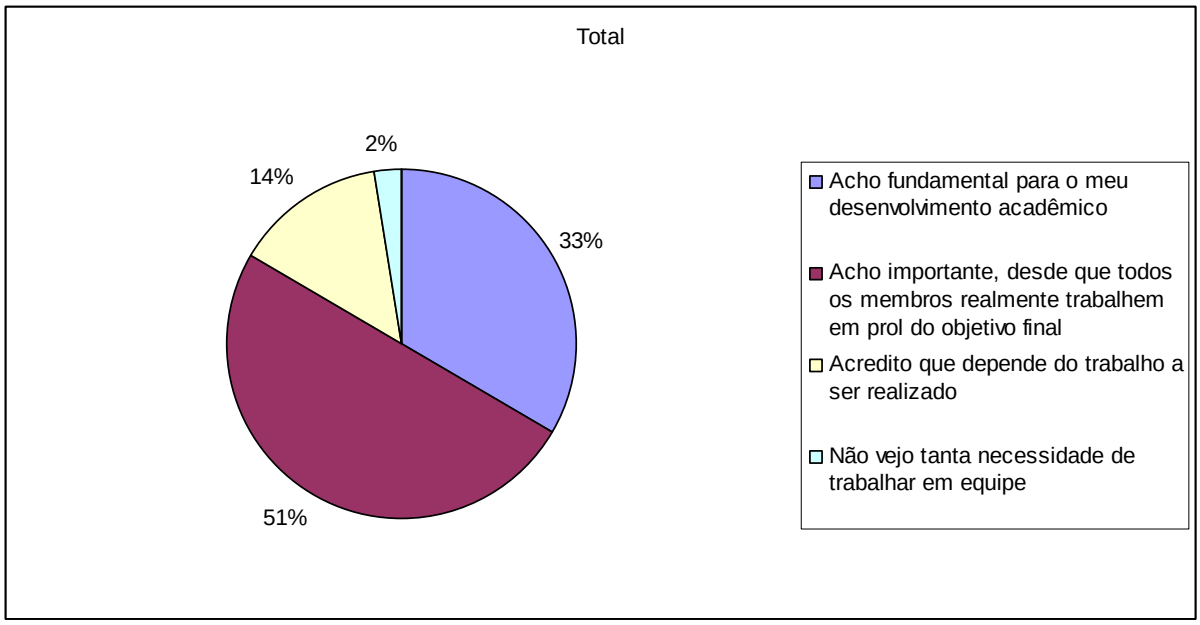


Figura 18: A importância do trabalho em equipe

Os estudantes foram questionados a respeito do número ideal de componentes em uma equipe para que pudessem realizar um bom trabalho em equipe. A tabela 2 apresenta esses dados

Tabela 2: Número ideal de integrantes de uma equipe

Três	11
Quatro	26
Cinco	4
Mais de cinco	1
Total	42

A respeito das dificuldades encontradas durante a realização de trabalhos em grupo, 31 estudantes apontaram a incompatibilidade de horários entre os membros da equipe como um grande empecilho às suas atividades. A ausência de divisão de tarefas e a dificuldade para compartilhar conteúdo também foram considerados fatores bastante relevantes. Cada estudante pode elencar mais de um fator, portanto, a totalização dos dados gera um número maior que o de participantes da pesquisa.

Na Figura 19 há um gráfico ilustrativo para essas informações.

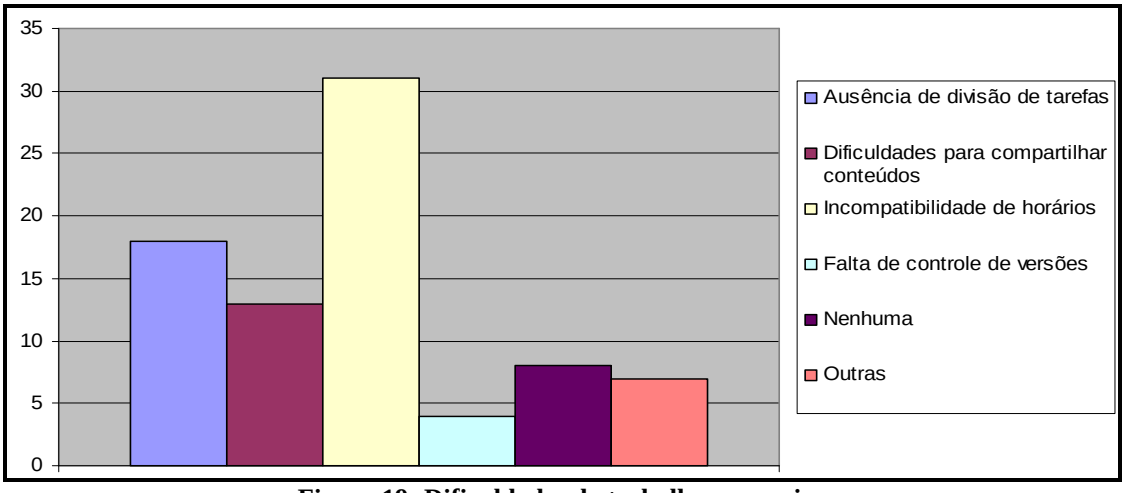


Figura 19: Dificuldades do trabalho em equipe

Dos quarenta e dois alunos que participaram da pesquisa, 27 já utilizaram alguma ferramenta colaborativa para a realização de trabalhos acadêmicos. Dos alunos matriculados entre o I e o III semestres, metade utiliza essas ferramentas. Já para alunos que cursam a partir do IX semestre, essa utilização sobe consideravelmente. Entre os alunos questionados, 14

utilizam essas ferramentas, contra apenas 4 que não fazem o uso. É possível concluir que com o decorrer da graduação, os alunos buscam novos mecanismos para auxiliá-los na vida acadêmica.

A tabela 3 traz a quantidade de alunos que utilizam e não utilizam as ferramentas em questão (agrupados por faixa de semestre)

Tabela 3: Quantidade de alunos que já utilizaram ferramentas colaborativas.

Não	Entre o I e o III	4
	Entre o IV e o VI	3
	VII ou VIII	3
	A partir do IX	5
Subtotal (Não)		15
Sim	Entre o I e o III	4
	Entre o IV e o VI	6
	VII ou VIII	3
	A partir do IX	14
Subtotal (Sim)		27
Total		**42**

De acordo com os dados coletados, as ferramentas mais utilizadas pelos alunos são os programas de mensagem instantânea e grupos. Essas informações estão representadas no gráfico da Figura 20

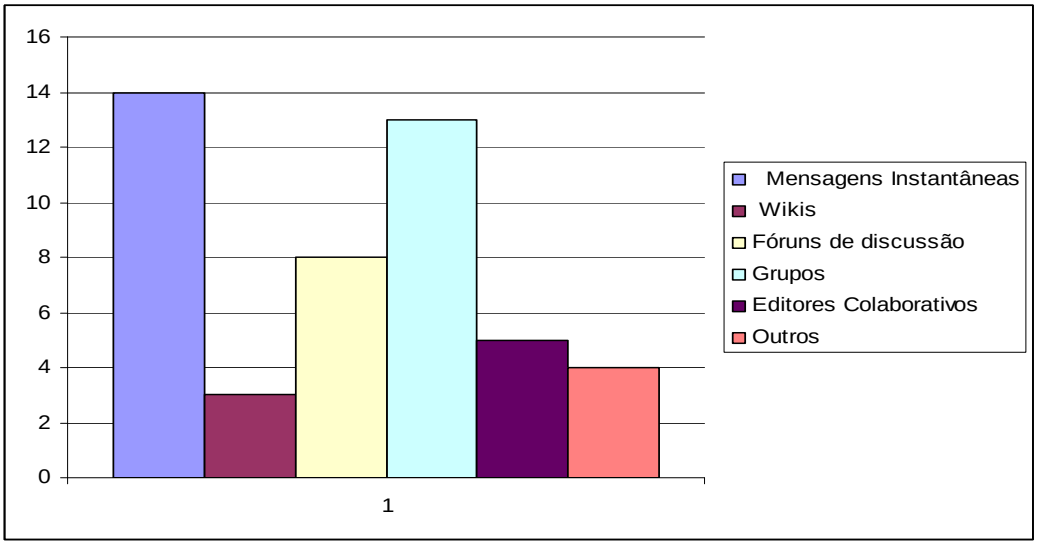


Figura 20: Tipos de ferramentas colaborativas utilizadas pelos alunos

Por fim, os alunos expressaram sua opinião a respeito da utilização dessas ferramentas. Cinquenta e nove por cento dos alunos consideram bastante útil o uso de tais ferramentas, 4% consideram excelente, já 37% têm a opinião que elas são boas, mas poderiam ser melhores para o desempenho acadêmico. Nenhum estudante teve uma opinião negativa a esse respeito. A Figura 21 ilustra esses dados.

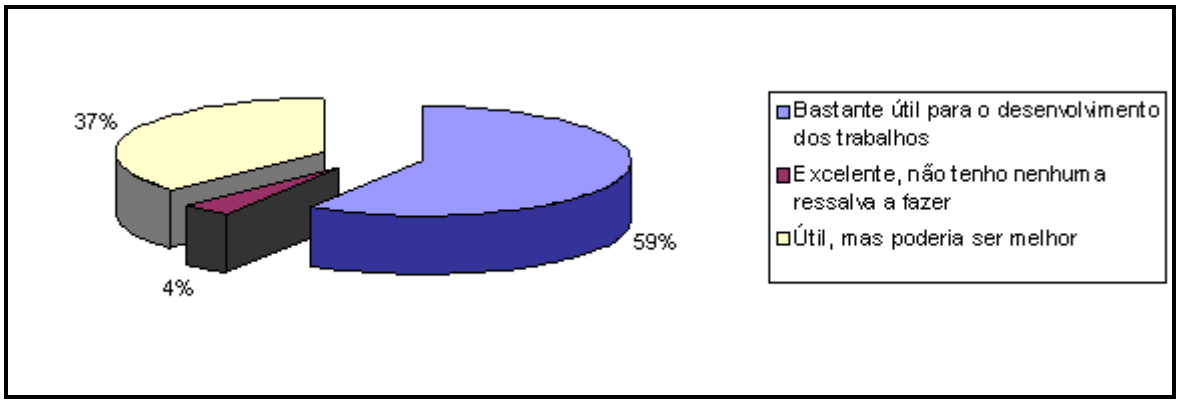


Figura 21: Opinião a respeito da utilização das ferramentas colaborativas

4.3 CollabPro

Após uma breve explanação a respeito dos ADC's e a análise dos dados extraídos dos questionários, foi elaborada a metodologia CollabPro. A Figura 22 mostra o ciclo básico da metodologia e a seguir são descritas as suas etapas.

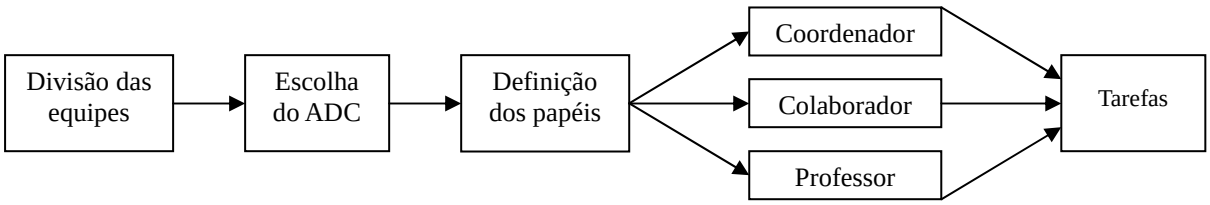


Figura 22: Ciclo Básico da ColabPro

4.3.1 Divisão das equipes

Ao início da disciplina Engenharia de Software os alunos formarão equipes com quatro integrantes, pois, conforme visto na seção 4.2, esse é considerado o número ideal de membros de um grupo de trabalho.

4.3.2 Escolha do ADC

Ainda na disciplina Engenharia de Software, os alunos, em conjunto com o professor da disciplina, podem decidir qual o ambiente eles julgam mais adequado para a realização das suas atividades. Para este foi escolhido o INRIA por apresentar uma interface natural e intuitiva e oferecer vários recursos essenciais à comunicação.

É muito importante a escolha do ambiente, pois ele será o canal principal para a aplicação do Modelo 3C de Colaboração, por isso é importante que essa escolha seja feita analisando as funcionalidades e deficiências de cada um.

4.3.3 Definição de papéis

Essa metodologia conta com a presença de três papéis:

- **Coordenador –** responsável por incluir projetos, colaboradores e tarefas no sistema;
- **Colaborador –** responsável por verificar as tarefas pendentes, realizar atividades e iniciar fóruns;
- **Professor –** periodicamente irá acessar o ambiente para verificar as atividades realizadas pelos alunos para se inteirar da expectativa de aprendizado deles.

Dentre os quatro componentes da equipe um será designado coordenador e os outros três serão colaboradores. Convém ressaltar que todos os alunos serão avaliados pelo professor de forma igualitária e os papéis são definidos de modo a facilitar a interação no sistema.

Outra ressalva importante se refere ao papel do professor. Diferentemente da EaD (Educação a Distância), ele não será responsável por inserir material didático no ambiente. Sua participação consistirá em visualizar as atividades realizadas pelos discentes para obter um *feedback* da aprendizagem da turma. É facultado ao professor participar dos fóruns ou listas de discussão presentes no ADC.

4.3.4 Identificando os elementos 3C

O INRIA oferece suporte a vários recursos colaborativos. A comunicação é feita através das listas de discussão. Esse ambiente, infelizmente, não oferece nenhum programa de mensagens instantâneas, então, para aumentar a comunicação, os alunos precisam utilizar um software dessa natureza, tais como *MSN Messenger*, *GoogleTalk* ou similares.

Não é necessária a presença de todos os componentes da equipe durante as conversas on-line, mas é importante que as decisões tomadas sejam relatadas no ambiente, para que todos possam ter acesso ao que foi discutido.

Ao submeter um projeto no INRIA o coordenador deve se atentar a duas configurações:

Escolher um repositório para controle de versão – O sistema oferece duas opções de controle de versão: o *Subversion* (controla os arquivos em um servidor centralizado) e o *Git* (os arquivos ficam armazenados de forma distribuída).

Definir a privacidade do projeto – o INRIA permite que o usuário determine se o projeto é público ou privado. No primeiro caso, o projeto ficará disponível para qualquer usuário do site. O ideal é selecionar a segunda opção, pois assim apenas usuários autorizados poderão contribuir com o projeto.

Além disso, a fase da coordenação se estende a incluir colaboradores e gerenciar as tarefas. Sendo esta uma ação bastante recorrente, devido ao surgimento de novas atividades, o coordenador sempre estará utilizando o sistema para a divisão de tarefas. Vale ressaltar, novamente, que o coordenador é, antes de tudo, um aluno, sendo assim ele também será responsável pela realização de algumas tarefas.

A realização de tarefas em grupo é uma ação inerentemente cooperativa. Quando os usuários utilizam os fóruns também estão cooperando. A associação desses três elementos colaborativos auxilia os alunos a obterem um maior proveito das suas atividades em equipe.

4.3.5 Aplicação da metodologia

Foram escolhidas seis disciplinas do currículo do curso de Ciência da Computação. A proposta é que os alunos comecem a utilizar o ambiente já na primeira disciplina (nesse caso,

Engenharia de Software) e a partir daí continuem até o desenvolvimento do produto de software final.

A CollabPro prevê, basicamente, que os alunos utilizem a colaboração como princípio chave para realizar as atividades em equipe. Como não é possível prever a sequência na qual os alunos cursarão as disciplinas optativas do perfil, serão sugeridos alguns exemplos de atividades que podem ser feitas em cada disciplina utilizando o INRIA, baseando-se nas ementas do anexo II. São mostradas na tabela 4 algumas dessas sugestões.

Tabela 4: Exemplo de atividades que podem ser realizadas de forma colaborativa

Disciplinas	Atividades
Engenharia de Software	Discutir/debater os conceitos, Pesquisar metodologias aplicadas à Engenharia de Software e compartilhar os resultados.
Análise e Modelagem de Sistemas	Pesquisar modelos de projetos de sistemas, Escolher um projeto inicial para desenvolvimento de software. Desenvolver os diagramas do sistema.
Engenharia de Requisitos	Elicitar os requisitos do software (funcionais e não-funcionais) e gerenciá-los. Pesquisar e compartilhar informações a respeito da Engenharia de Requisitos.
Gerenciamento de Projetos	Dividir entre os alunos a pesquisa sobre ferramentas de gerenciamento de projetos. Criar e gerenciar projetos de software.
Desenvolvimento de Software	Desenvolver o projeto de software de acordo às etapas estudadas, Utilizar o sistema de controle de versões nos arquivos de documentação e codificação do sistema. Elaborar artefatos do sistema.
Qualidade de Software	Discutir a respeito das normas de qualidade de software. Analisar a qualidade dos softwares desenvolvidos e adequá-los às normas estudadas.

É interessante que em todas as disciplinas os alunos utilizem os fóruns e as listas de discussão auxiliar no processo de aprendizagem do conteúdo.

4.3.6 Dificuldades

Algumas dificuldades podem aparecer durante a aplicação dessa metodologia. Caso o coordenador da equipe não esteja desempenhando a sua função de forma satisfatória, o resultado final do grupo certamente será prejudicado. Para lidar com isso, os alunos precisam constantemente analisar a coordenação e/ou transferir esse papel para outro estudante. Essa mudança na coordenação ajuda a distribuir responsabilidades iguais entre a equipe.

Outra dificuldade pode aparecer após o término de uma disciplina e início de outra. Não se pode garantir que os grupos continuarão com a mesma formação, devido a alguns fatores:

- Relacionamento ruim entre os membros da equipe;
- Escolha de perfis diferentes;
- Reprovação na disciplina em curso;

Independente da situação ocorrida, os alunos do novo grupo devem avaliar os projetos já existentes (e arquivados no ambiente) e decidir quais serão continuados. Os estudantes irão utilizar o conhecimento adquirido nas outras disciplinas para realizar os novos projetos.

Também pode acontecer de os alunos não gostarem do ambiente escolhido. Nesse caso, se houver um consenso, eles podem utilizar outro ambiente, fazendo as adaptações necessárias.

As ilustrações referentes às atividades de projetos no INRIA estão nos anexos III.

5 – CONCLUSÃO E TRABALHOS FUTUROS

5.1 Conclusão

Os sistemas colaborativos estão cada vez mais presentes no cotidiano das pessoas. Deve-se isso a vários fatores, porém um dos mais relevantes é a incompatibilidade de horários entres os indivíduos.

Esse trabalho permitiu a confirmação de que os estudantes utilizam *groupwares* para realizar suas atividades acadêmicas, porém necessitam de um mecanismo que otimize o uso desses softwares para fins acadêmicos. A CollabPro fica como sugestão para que sejam aplicados os conceitos de colaboração do Modelo 3C por meio de um ambiente colaborativo.

Os objetivos almejados nessa monografia não foram totalmente contemplados, pois é necessário aplicar a metodologia, em pelo menos uma turma, para que possa analisar seus resultados. A construção da metodologia sugere que a comunicação entre a equipe seja otimizada além de possibilitar maios interação entre alunos e professores. A utilização de um ambiente colaborativo facilita a reutilização de trabalhos já desenvolvimentos pelos discentes.

A principal dificuldade encontrada durante a elaboração dessa monografia foi a deficiência de sistemas colaborativos para fins acadêmicos. Foram encontrados alguns trabalhos que propunham *groupwares* acadêmicos, mas devido à sua complexidade de implementação, poucos são os sistemas finalizados. Por isso optou-se por utilizar o INRIA, um ambiente de fácil manipulação, sem necessidade de instalação e customização, facilitando assim a utilização do mesmo pelos discentes e docentes do curso.

Pode-se concluir que a crescente utilização de softwares colaborativos tem impulsionado o desenvolvimento de novas formas de colaboração o que abrange o universo acadêmico. São necessários maiores estudos e investimentos na criação de soluções voltadas ao ensino, pois é uma área que, apesar de custosa e complexa, auxilia grandemente os alunos, tanto possibilitando realizar suas tarefas, como estudando conceitos e detalhes da sua implementação.

5.2 Trabalhos Futuros

Uma sugestão para dar continuidade a esse estudo consiste em uma análise da viabilidade da aplicação dessa metodologia nos outros perfis do curso de Ciência da Computação da UESB. Nesse caso é necessário analisar se os softwares aqui citados contemplam a realidade do perfil em questão.

Continuar pesquisando a existência de *groupwares*, especialmente os direcionados ao âmbito acadêmico, também é uma maneira de prosseguir com os estudos aqui iniciados. O conhecimento sobre novas ferramentas e ambientes colaborativos é importante não só para a utilização dos mesmos nas atividades curriculares, como também para que os alunos possam ter mais familiaridade com detalhes técnicos da implementação de um software dessa natureza.

Também seria interessante a criação de uma incubadora de projetos utilizando uma customização do Gforge. Consistiria na utilização dos recursos disponibilizados pelo ambiente para moldar um local onde os alunos pudessem iniciar seus projetos, principalmente os de desenvolvimento de software.

Por fim, poderia ser feito um estudo mais aprofundado a respeito de todas as etapas e peculiaridades que envolvem o desenvolvimento de *groupware* para posteriormente desenvolver um *groupware* para fins acadêmicos. Essa sugestão se adequaria a um projeto de pesquisa contínuo, o que criaria uma boa base de conhecimento no assunto para que novos integrantes criem e aprimorem mais funcionalidades nesse sistema.

REFERÊNCIAS

ARANGO, G., PRIETO, R. *Domain Analysis Concepts and Research Directions*, in: Domain Analysis And Software System Modeling, 1a ed. California, IEEE Computer Society Press Tutorial, pp. 09-25, 1991.

ASTIVERA, A. *Metodologia da pesquisa científica* (Trad. de Maria Helena Guedes Crespo e Beatriz Marques Magalhães). 6. ed. Porto Alegre: Globo, 1980.

BECKER, K. & ZANELLA, A.N. *A Cooperation Model for Teaching/Learning Modeling Disciplines,* International Workshop on *Groupware* (CRIWG 1998), Brasil.

BENBUNAN-FICH, R. & HILTZ, S.R. (1999), *Impacts of Asynchronous Learning Networks on Individual and Group Problem Solving:* A Field Experiment, Group Decision and Negotiation, 8, 409-426

BOOCH, G., RUMBAUGH, J. & JACOBSON, I. *UML:* Guia do Usuário. Rio de Janeiro: Campus, 2000.

BRNA, P. *Modelos de colaboração.* Revista Brasileira de Informática e Educação, 3, pp. 1-15, 1998.

COSTA, J.M., FEITOSA, R.M., SOUZA, C.R. *RaisAware*: Uma ferramenta de auxílio à Engenharia de Software Colaborativa baseada em Análises de Dependências. Simpósio Brasileiro de Sistemas Colaborativos (SBSC). Vila Velha- ES, 2008

DEMO, P. *Pesquisa e construção do conhecimento*: metodologia científica no caminho de Habermas. Rio de Janeiro: Tempo Brasileiro, 1994.

DEWAN, P. (1998) *Architectures for Collaborative Applications.* In M.Beaudouin- Lafon (Ed.), Computer Supported Cooperative Work (CSCW) (7 ed., pp. 169- 194). John Wiley & Sons Ltd.

FALBO, R. A. *Engenharia de software.* 01 mar. 2005, 30 jun. 2005. 99 p. Notas de Aula. UEFS.

FUKS, H. RAPOSO, A.B. & GEROSA, M.A. (2002) *Engenharia de Groupware: Desenvolvimento de Aplicações Colaborativas.* XXI Jornada de Atualização em Informática, Anais do XXII Congresso da Sociedade Brasileira de Computação. V1. Cap 3, ISBN 85-88442-24-8, pp 89-128.

FUKS, H., RAPOSO, A.B. & GEROSA, M.A. *Do Modelo de Colaboração 3C à Engenharia de Groupware,* IX Simpósio Brasileiro de Sistemas Multimídia e Web – Webmidia 2003, Trilha especial de Trabalho Cooperativo Assistido por Computador, 03-06 de Novembro, Salvador-BA, pp. 445-452.

GEROSA, M.A., Pimentel, M.G., Filippo, D., Barreto, C.G., Raposo, A.B., Fuks, H. & Lucena, C.J.P. *Componentes Baseados no Modelo 3C para o Desenvolvimento de Ferramentas Colaborativas*, Anais do 5º Workshop de Desenvolvimento Baseado em Componentes - WDBC 2005, 07-09 de Novembro, Juiz de Fora-MG, ISBN 85-88279-47-9, pp. 109-112. Disponível em http://www.les.inf.puc-rio.br/*groupware*

GEROSA, M. A. *Desenvolvimento de groupware componentizado com base no modelo 3C de colaboração.* Rio de Janeiro : PUCRio. Departamento de Informática, 2006.

Gforge Collaborative Development Environment. Disponível em http://gforge.org/gf/. Acessado em 02/12/2010.

INRIA Gforge. Disponível em http://gforge.inria.fr. Acessado em 03/12/2010.

JOHNSON, R.E. *Frameworks = Components + Patterns*, Communications of the ACM, USA, Vol. 40, N. 10, p. 39-42, 1997.

LibreSource. Disponível em http://dev.libresource.org. Acessado em 01/12/2010.

MANGAN, M. A. S. *Uma Abordagem para o Desenvolvimento de Apoio à Percepção em Ambientes Colaborativos de Desenvolvimento de Software.* Rio de Janeiro: COPPE/UFRJ, 2006.

MENDES, Antonio. *Arquitetura de software:* Desenvolvimento orientado para arquitetura. Rio de Janeiro: Campus, 2002.

MOECKEL, A. *Cscw*: Conceitos e aplicações para cooperação. Curitiba: Cefet-Pr, 2003.

PIANCASTELLI CH, Faria HP, Silveira MR. *O trabalho em equipe*. In: Santana JP, organizador. Organização do cuidado a partir de problemas: uma alternativa metodológica para a atuação da equipe de saúde da família. Brasília: OPAS/Representação do Brasil; 2000. p 45-50.

PIMENTEL, M. *RUP-3C-Groupware*: um processo de desenvolvimento de groupware baseado no Modelo 3C de Colaboração, Tese de Doutorado, Departamento de Informática, PUC-Rio, Rio de Janeiro, RJ, 2006.

PIMENTEL, M., GEROSA, M.A., FILIPPO, D., RAPOSO, A., FUKS, H. & LUCENA, C.J.P. *Modelo 3C de Colaboração no Desenvolvimento de Sistemas Colaborativos*. Anais do III Simposio Brasileiro de Sistemas Colaborativos, Natal - RN, 20 a 22 de Novembro de 2006. ISBN 85-7669-097-7. Porto Alegre: SBC, 2006. pp. 58-67. Disponível em http://*groupware*.les.inf.puc-rio.br

PRESSMAN, Roger S. *Engenharia de Software*. 5. ed. São Paulo: Makron Books, 1995.

ProjectPier. Disponível em http://www.projectpier.org. Acessado em 01/12/2010.

SOMMERVILLE, I. *Engenharia de Software*, 8a. edição. São Paulo: Pearson Addison-Wesley, 2007

TEIXEIRA, B. & CHAGAS, E.F. *Co-Autoria: Avaliação e Proposta de Requisitos para Ferramentas Segundo o Modelo 3C.* Workshop de Informática na Escola, Congresso da Sociedade Brasileira de Computação 2005.

TIBERTI, A. J. *Desenvolvimento de um sistema gerenciador de fluxo de trabalho para um ambiente de suporte a atividades de engenharia.* São Carlos, 1996. Dissertação (Mestrado em Engenharia) – EESC, USP.

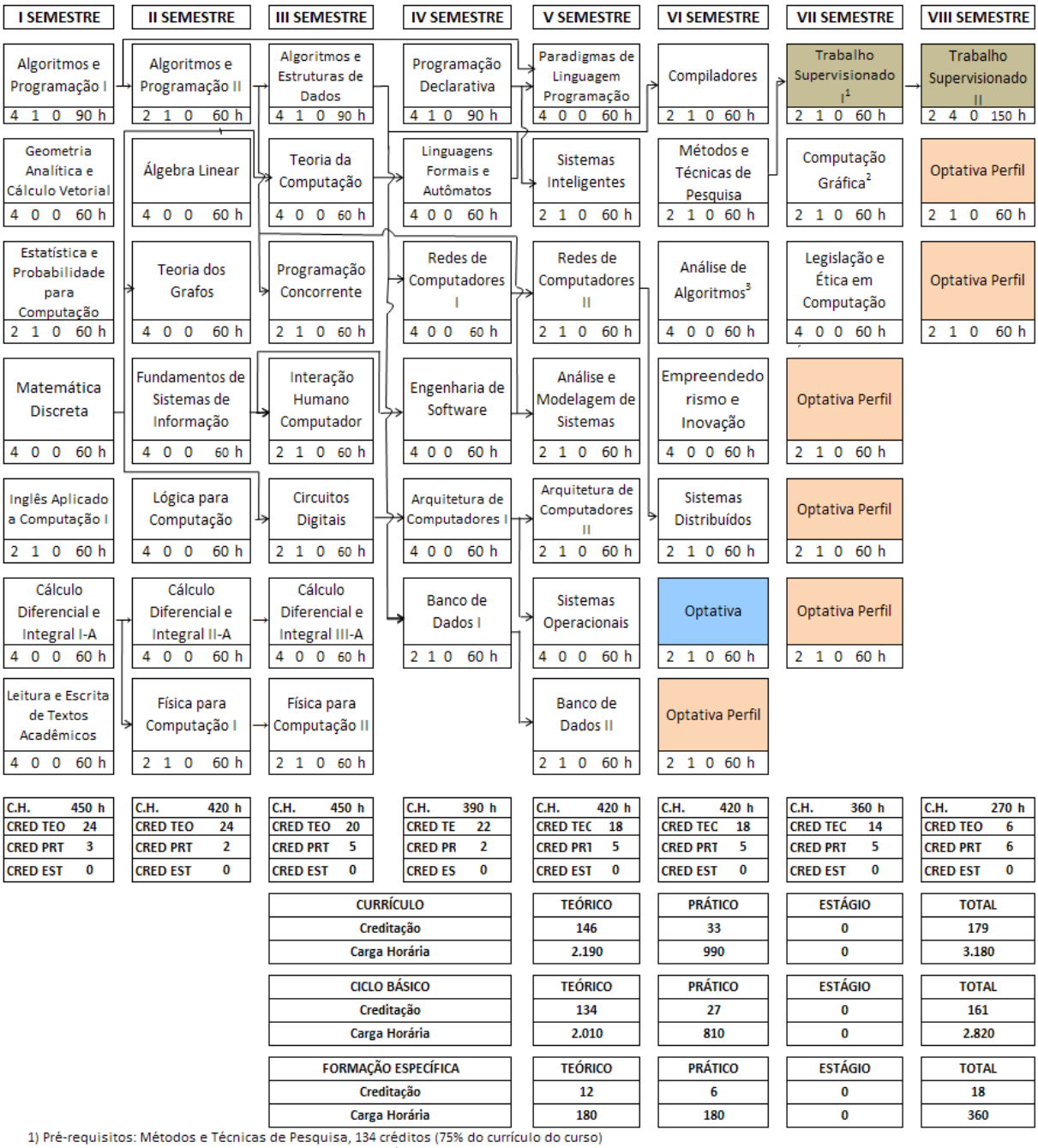
TIETZE, D.A., A *Framework for Developing Component-based Co-operative Applications*. PhD Dissertation, Computer Science, Technischen Universität Darmstadt, Germany, 2001.

Trac. Disponível em http://trac.edgewall.org/. Acessado em 01/12/2010.

ZLATANOV, T. *Git para usuários do Subversion*. Disponível em http://www.ibm.com/developerworks/br/linux/library/l-git-subversion-1/. Acessado em 15/12/2010.

ANEXOS

ANEXO I: Fluxograma do Curso de Ciência da Computação (após a reforma curricular)



1) Pré-requisitos: Métodos e Técnicas de Pesquisa, 134 créditos (75% do currículo do curso)
2) Pré-requisitos: Geometria Analítica e Cálculo Vetorial, Álgebra Linear
3) Pré-requisitos: Algoritmos e Estruturas de Dados, Matemática Discreta

ANEXO II: Ementas das disciplinas escolhidas para o desenvolvimento da metodologia

Engenharia de Software

Pré-requisito: Fundamentos de Sistemas de Informação.
Introdução a Engenharia de Software. Processo de desenvolvimento de software. Ciclo de vida de Software. Planejamento e gerenciamento de software. Engenharia de Requisitos. Qualidade de Software. Projeto e arquitetura de software. Verificação, Validação e Teste. Manutenção e evolução do software. Metodologias aplicadas à engenharia de software.

Análise e Modelagem de Sistemas

Pré-requisito: Engenharia de Software e Algoritmos e Programação II.
Uso de modelos, metodologias, técnicas e ferramentas de análise e projeto de sistemas (paradigma estruturado e orientado a objeto).Técnicas e ferramentas para análise de sistemas.

Engenharia de Requisitos

Pré-requisito: Engenharia de Software.
Elicitação de requisitos, identificação das fontes de informação; técnicas de elicitação; modelagem; técnicas de modelagem; análise de requisitos; validação e verificação; gerência de requisitos; certificação e padrões internacionais; ferramentas.

Gerenciamento de Projetos

Pré-requisito: Análise e Modelagem de Sistemas.
Determinação do Escopo de Projetos. Ciclo de Vida de Projetos. Fases de um Projeto. Modelo de Processos do Planejamento e Gestão de Projetos. Artefatos da Gerência de Projetos. Certificação da Qualidade do Processo e do Produto. Gerenciamento de Custos, Riscos, Prazos e Comunicações. Ferramentas de Software para a Gestão de Projetos. Estudo de casos.

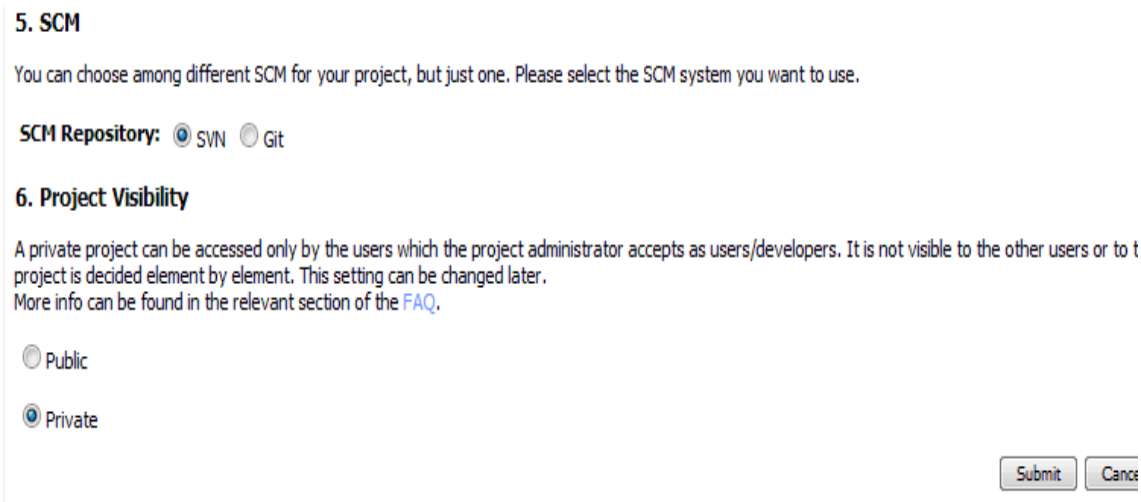
Desenvolvimento de Software

Pré-requisito: Algoritmos e Programação II, Análise e Modelagem de Sistemas
Principais teorias, métodos, técnicas e ferramentas associadas ao projeto de software. Consolidação, através de um projeto real de uma dada organização, dos conceitos abordados no currículo básico do curso, via desenvolvimento de sistemas em grupo, sob a orientação do professor.

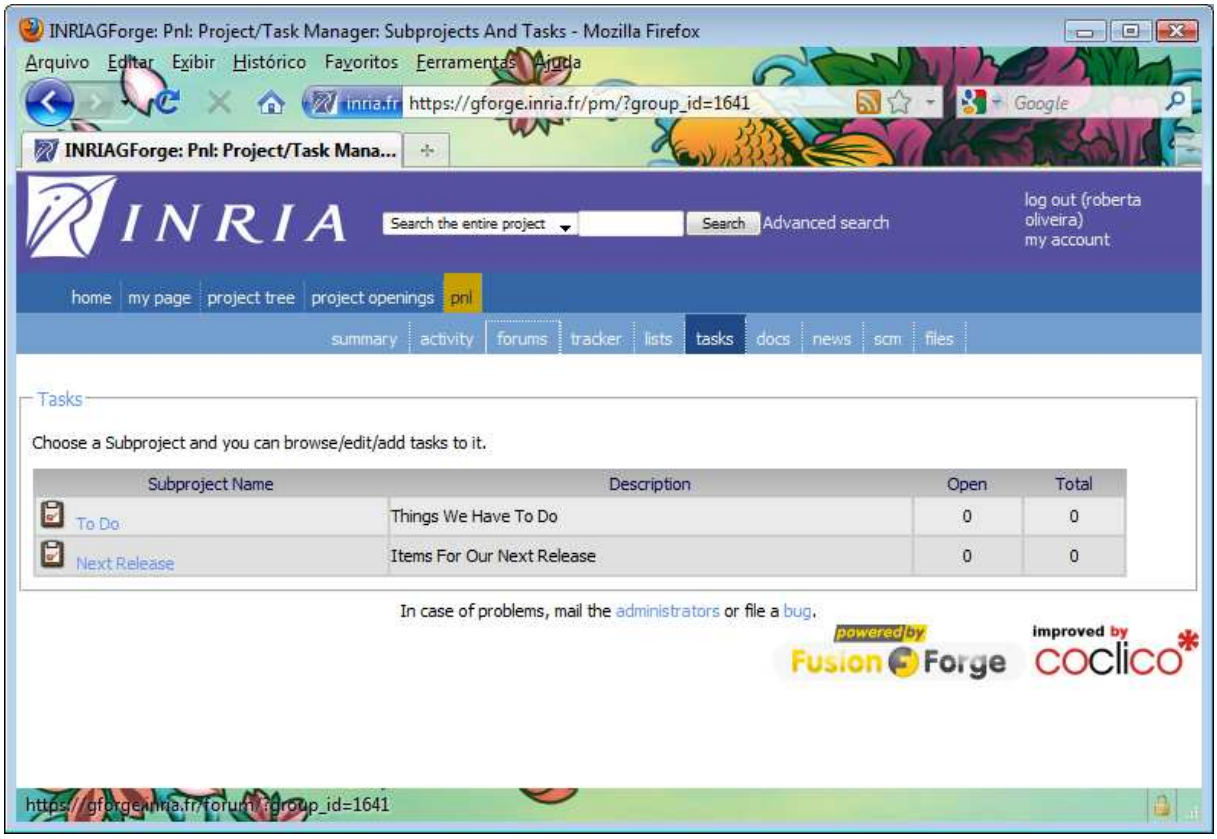
Qualidade de Software

Pré-requisito: Engenharia de Software
Qualidade de Software. Programas de Qualidade e Métricas. Normas de Qualidade. Ambientes de Desenvolvimento. Técnicas de projeto, construção, seleção e o uso de Ambientes e Ferramentas de Desenvolvimento.

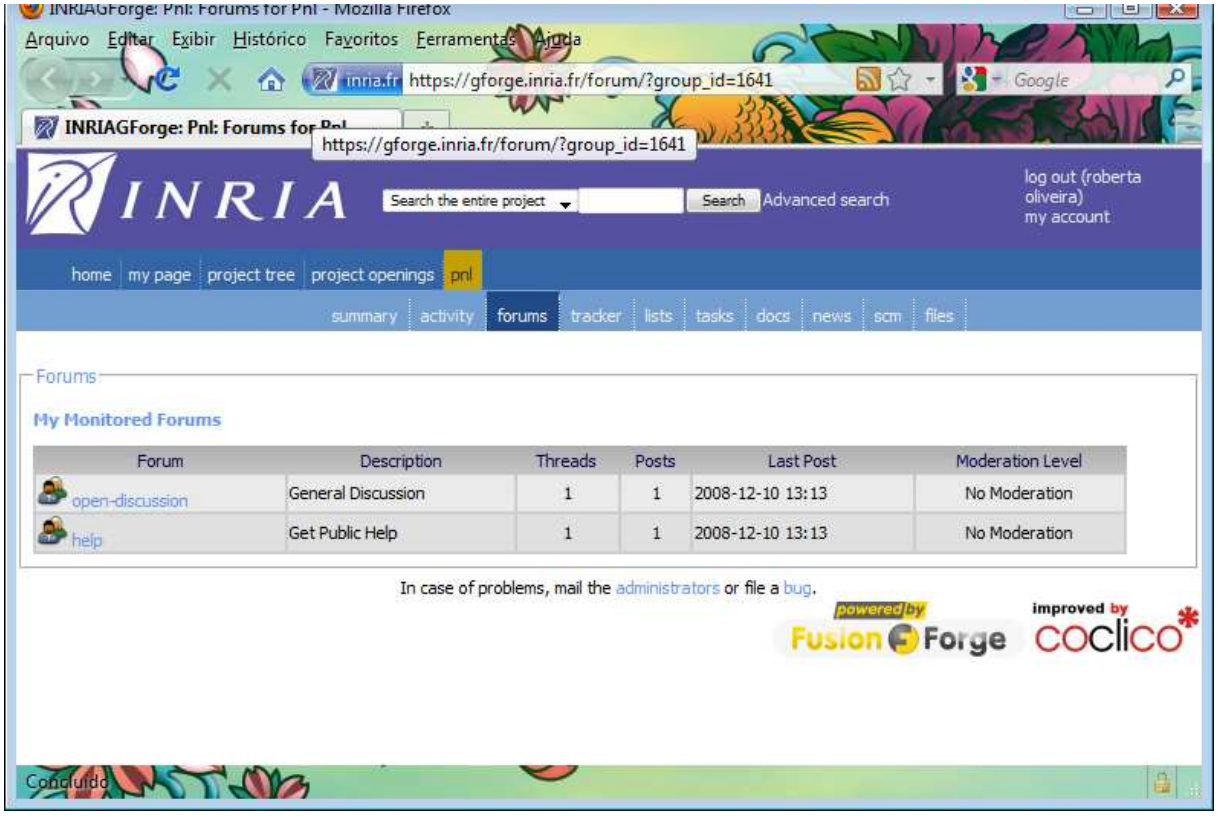
ANEXO III: Telas do INRIA



Cadastrando um projeto. Escolhendo a repositório de controle de versão (SVN-Subversion ou Git) e configurando a privacidade do projeto (*public* ou *private*).



Descrição de tarefas


INRIAGForge: Pnl: Forums for Pnl - Mozilla Firefox
Arquivo Editar Exibir Histórico Favoritos Ferramentas Ajuda
https://gforge.inria.fr/forum/?group_id=1641
INRIAGForge: Pnl: Forums for Pnl
https://gforge.inria.fr/forum/?group_id=1641
INRIA
Search the entire project
Search
Advanced search
log out (roberta oliveira)
my account
home
my page
project tree
project openings
pnl
summary
activity
forums
tracker
lists
tasks
docs
news
scm
files
Forums
My Monitored Forums
Forum
Description
Threads
Posts
Last Post
Moderation Level
open-discussion
General Discussion
1
1
2008-12-10 13:13
No Moderation
help
Get Public Help
1
1
2008-12-10 13:13
No Moderation
In case of problems, mail the administrators or file a bug.
powered by FusionForge
improved by COCLICO
Concluído

Fóruns

APÊNDICE I

Questionário

Questionário direcionado aos alunos de Ciência da Computação da UESB. As respostas serão utilizadas no Trabalho de Conclusão de Curso de Roberta Mercia R. de Oliveira.

---

1 – Atualmente, você cursa qual semestre?

- o Entre o I e o III;
- o Entre o IV e o VI;
- o VII ou VIII;
- o A partir do IX

2 – Qual sua opinião a respeito do trabalho acadêmico em equipe?

- o Acho fundamental para o meu desenvolvimento acadêmico;
- o Não vejo tanta necessidade de trabalhar em equipe;
- o Acho importante, desde que todos os membros realmente trabalhem em prol do objetivo final;
- o Acredito que depende do trabalho a ser realizado

3 – Na sua opinião, de um modo geral, qual o número máximo de componentes que uma equipe precisa ter para atingir êxito nas atividades acadêmicas?

- o Três
- o Quatro
- o Cinco
- o Mais de cinco

4 – Qual(is) dificuldade(s) você já teve ao desenvolver trabalhos em grupo?

- □ Incompatibilidade de horários entre os membros;
- □ Ausência de uma divisão de tarefas;
- □ Dificuldades para compartilhar conteúdo;
- □ Falta de controle nas atualizações das versões;
- □ Nenhuma;
- □ Outras.

5 – Você já utilizou alguma ferramenta ou ambiente colaborativo para fins acadêmicos?

- o Sim
- o Não

6 – Qual dessas ferramentas colaborativas você já utilizaou? (Responda APENAS se houver respondido SIM à questão 5)

- □ Mensagens Instantâneas;
- □ Wikis;
- □ Fóruns de discussão;
- □ Grupos;
- □ Editores Colaborativos;
- □ Outros

7 – O que achou da utilização dessas ferramentas? (Responda APENAS se houver respondido SIM à questão 5)

- o Bastante útil para o desenvolvimento dos trabalhos;
- o Útil, mas poderia ser melhor;
- o Excelente. Não tenho nenhuma ressalva a fazer;
- o Não me ajudou significativamente