UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO
DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO INDUSTRIAL

SUPORTE À COLABORAÇÃO EM REDES P2P

Autor
Dalton Lopes Martins

Orientador
Prof. Dr. Ivan Luiz Marques Ricarte

Banca Examinadora:
- Prof. Dr. Ivan Luiz Marques Ricarte
- Prof. Dr. Léo Pini Magalhães
- Prof. Dra. Heloísa Vieira da Rocha
- Prof. Dra. Lúcia Isaltina Clemente Leão

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica, área de concentração: Engenharia de Computação.

Campinas, 20 de Dezembro de 2004

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

M366s

Martins, Dalton Lopes
Suporte à colaboração em redes P2P / Dalton Lopes Martins. --Campinas, SP: [s.n.], 2004.

Orientador: Ivan Luiz Marques Ricarte
Dissertação (Mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Grupos de trabalho virtual. 2. Arquitetura de redes de computador. 3. Java (Linguagem de programação de computador). 4. Ensino a distância. I. Ricarte, Ivan Luiz Marques. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Título em Inglês: Collaborative support in P2P networks
Palavras-chave em Inglês: Virtual work teams, Network archictures Computer, Java (Computer program language) e Distance education
Área de concentração: Engenharia de Computação
Titulação: Mestre em Engenharia Elétrica
Banca examinadora: Léo Pini Magalhães, Heloísa Vieira da Rocha e Lúcia Isaltina Clemente Leão
Data da defesa: 20/12/2004

Agradecimentos

Este é certamente o resultado de um trabalho que se iniciou em uma idéia há algunas anos atrás, onde o término da graduação e o início da vida profissional era algo ainda bastante nebuloso. Pude dividir essa idéia com muitas pessoas, mas algumas realmente deram todas as condições para que ela pudesse se desenvolver e se chegar ao seu objetivo final. Não poderia deixar de citá-las nesse momento.

Certamente, meus pais, Edgard e Ivone, que me deram sempre todos os recursos possíveis e imagináveis para que eu pudesse desenvolver meus estudos e atingisse minhas metas. O meu grande irmão, André, que me incentivou e me deu o apoio necessário para vir para esta universidade e iniciar uma nova etapa em minha vida.

Alguns amigos, ou mais que amigos, que me acompanharam em muitos momentos da construção desse sonho e me alimentaram com referências, conversas e idéias: Leonardo Mecchi, Fábio Henrique, Bruno Drago, Ricardo Freire, Munir Younes e Glauco Paiva.

Aos companheiros de projeto MetaReciclagem e das utopias de projeto Metáfora que me auxiliaram nas primeiras discussões sobre redes, colaboração e seus desdobramentos sociais: Hernani Dimantas, Felipe Fonseca e Daniel Pádua.

Ao meu orientador por ter toda a liberdade de criação e trabalho em cima de minhas idéias.

E por fim, a minha querida Flávia, que me deu todo o apoio quando muitas das coisas pareciam simplesmente não dar certo.

Resumo

O suporte à colaboração e as redes P2P *(peer-to-peer)* tem sido dois assuntos que, por muitos momentos, se cruzam no estudo e na pesquisa do uso das novas tecnologias em ambientes colaborativos de aprendizado. O presente estudo trata das motivações sociais e técnicas para o uso das redes P2P como um suporte ao trabalho colaborativo entre grupos. Inicialmente, realizamos um estudo das características de uso e da mudança de paradigmas de trabalho em rede que o P2P nos fornece. Como base de aplicação prática deste trabalho, também apresentamos um estudo de protocolos e sistemas computacionais já existentes que tratam, de alguma forma, da questão da colaboração através de redes P2P. Por final, com base no estudo apresentado, elaboramos a especificação e a implementação de um sistema de colaboração, o SemiCode, de forma a validar algumas das idéias desenvolvidas neste trabalho, além de apresentarmos algumas soluções onde o P2P pode atuar em conjunto com sistemas cliente-servidor.

Abstract

The support to the collaboration and P2P network are two related subjects in the study and research involving new technologies for collaboratives learning environments. The present study deals with social motivations and techniques for the use of P2P nets as a support to the collaborative work. Initially, we carry through a study of the characteristics of use and the change of paradigms of work in net P2P supplies. Supporting the development of a application in this work, also we present a study of protocols and existing computational systems that, in some way, deal with collaboration through P2P nets. Finally, based on this study, we elaborate the specification and the implementation of a collaborative system, the SemiCode, wich validates some of these ideas. We also present some solutions based on the integra from between P2P and client-server systems.

Índice de Figuras e Tabelas

Capítulo 2 – Fundamentação Téorica

Capítulo 3 – O Sistema de Colaboração em P2P: SemiCode

Capítulo 4 – Tecnologia e Implementação do SemiCode

Capítulo 5 – O SemiCode e Arquitetura Cliente/Servidor

Capítulo 1 – Introdução

Iniciar um estudo sobre ambientes colaborativos de ensino mediados pela tecnologia exige, se não quisermos apenas ficar em generalidades devido a complexidade e variedade do tema, um recorte técnico com base no problema que se pretende resolver e a contextualização da tecnologia escolhida em meio a um cenário essencialmente multidisciplinar, que toca áreas como a pedagogia e a sociologia.

Construir esse recorte técnico passa necessariamente por compreender o processo de desenvolvimento da tecnologia e os desdobramentos de sua utilização dentro do contexto em que está inserida. Desta forma, analisando o cenário atual de aplicações voltadas para ambientes colaborativos de ensino podemos nos deparar com dois paradigmas relacionados basicamente à topologia de rede: estrutura cliente-servidor e *peer-to-peer.*

A estrutura cliente-servidor baseia-se no princípio de que diversas máquinas, chamadas de clientes, conectam-se a uma determinada máquina, chamada de servidor, para receber informações, enviar dados, enfim, possibilitando que toda a comunicação seja mediada pelo papel tecnológico do servidor. Essa estrutura tornou-se muito popular com a explosão do acesso doméstico a Internet na década de 90, com o conseqüente uso intensivo dos protocolos para modems, como o SLIP *(Serial Line IP)* e o PPP *(Point-to-Point Protocol),* e o aumento no uso de *firewalls* de segurança nas corporações (Oram, 2001, pág. 9). Dessa forma, o servidor adquiriu importância ímpar nas aplicações desenvolvidas sob essa lógica e tornou-se o "ponto de encontro" central dos clientes usuários dessas aplicações. A política de uso de uma rede cliente-servidor e a dinâmica de interação entre os clientes é, portanto, estabelecida pelo servidor, através de seus recursos técnicos disponíveis e sua capacidade de gerenciamento de usuários.

Da mesma forma que o crescimento da Internet no uso doméstico trouxe uma enorme quantidade de usuários para dentro da grande rede, ela também "encaixou a maior parte do tráfego no paradigma de curso descendente já

usado pela televisão e pelos jornais" (Oram, 2001, pág. 10). Esse paradigma de curso descendente está intimamente relacionado com os hábitos desenvolvidos a partir dessa tecnologia de uso da rede, ou seja, os clientes do sistema conectam-se aos servidores para obterem notícias, a cotação do dólar, os índices da bolsa, a previsão do tempo, os resultados imediatos do campeonato de futebol, simulando nesse acesso de curso descendente um uso da rede muito próximo ao assistir uma televisão, derivando daí a expressão, originada também nos anos 90, do "surfe na web" que "é uma derivação do surfe de canais – expressão que se impôs ao mundo com o surgimento dos controles remotos e da multiplicidade de canais por cabo em meados da década de 1980" (Johnson, 2001). Configura-se, de tal forma, um modelo de acesso a rede baseado no *download* de informações, onde o usuário, a máquina cliente, influencia muito pouco as políticas de comunicação e a troca efetiva de informações.

Tais hábitos de uso da rede derivam basicamente de algumas premissas técnicas que são levadas em consideração quando falamos em arquitetura cliente-servidor, como o fato de "a máquina do cliente da Web não precisa ter um endereço permanente ou conhecido. Ela não necessita de uma conexão contínua com a Internet. Não há a necessidade de conciliar vários usuários. Só precisa saber como fazer uma pergunta e ouvir a resposta" (Oram, 2001, pág.10). Essas premissas técnicas se refletem em uma tendência de padrão de conexão física a Internet que é o sistema ADSL *(Asymmetric Digital Subscribers Line)*, onde "uma instalação típica de ADSL ou modem a cabo oferece de três a oito vezes mais largura de banda quando obtém dados da Internet do que quando os envia para ela, favorecendo o uso do cliente em relação ao servidor" (Oram, 2001, pág.15). Essa limitição é praticamente transparente ao usuário da rede que tem como costume de uso o acesso baseado em *download*, onde ele é apenas cliente e recebe informações e não "serve" informações para a rede.

É, portanto, esse o cenário técnico e os hábitos de uso por ele despertados de uso da Internet até meados de 2000, período em que a ascensão das aplicações P2P (*peer-to-peer*) chama a atenção de estudiosos da rede e desenvolvedores de novas aplicações. É o período de explosão de uso

do Napster (famoso sistema de troca de arquivos entre usuários usando técnicas de redes P2P), sistema de troca de música entre usuários baseado num sistema híbrido cliente-servidor e P2P. O Napster funcionava basicamente da seguinte forma: usuários disponibilizavam uma lista de arquivos de música que desejavam compartilhar com outros usuários, essa lista era armazenada nos servidores do Napster e quando um usuário realizava uma busca por determinada música, essa busca ocorria dentro do escopo dos servidores; dado que a busca era bem sucedida, o sistema conectava diretamente o usuário detentor do arquivo e aquele que o desejava. Uma nova revolução de hábitos e costumes iniciava-se na Internet devido a um novo conjunto de premissas técnicas que possibilitavam colocar usuários em contato diretamente uns com os outros. Vem daí o fato de que "o Napster teve um impacto revolucionário devido a uma escolha básica de projeto: após a pesquisa inicial do material, os clientes se interconectaram e trocaram dados diretamente de seus discos rígidos de um sistema para o outro" (Oram, 2001, pág. VI).

A princípio, as questões envolvidas em torno dessas novas possibilidades de conexão poderiam apenas suscitar preocupações em torno de questões a respeito de direitos autorais, o que realmente ocorreu levando a uma das maiores disputas em tribunais da história da Internet que culminou com o fechamento do Napster nos modelos originais de uso. No entanto, havia muito mais por detrás das questões além de direitos autorais; havia uma nova possibilidade técnica de construção de redes que não fosse necessariamente mediada pelo papel de um servidor como "ponto de encontro" central na rede. Inúmeras aplicações, protocolos e serviços começaram a surgir com base nas idéias e nas discussões geradas pelo caso Napster.

A história do P2P não teve, no entanto, seu início com o Napster. Ela remonta aos tempos de formação da Internet, onde ainda não havia a necessidade de servidores centrais e a troca de informações se dava de ponto-a-ponto ou de par-a-par.

O Usenet News, um dos pioneiros da Internet no final da década de 60 quando era ainda chamada de ARPANET, "é um sistema que, não usando controle central, copiava arquivos entre computadores" (Oram, 2001, pág.5). Esse

sistema era baseado no protocolo UUCP (*Unix-to-Unix-Copy Protocol*), que basicamente ligava para uma máquina remota, conectava-se automaticamente, trocava informações e depois se desconectava. A idéia por trás do funcionamento desse protocolo é a mesma da troca de arquivos mediada pelo Napster: duas máquinas se conectam, trocam arquivos e depois a conexão é fechada.

O DNS (*Domain Name System*), que é o principal agente que possibilita o uso de nomes ao invés de números como endereços de sites na Web, é um sistema que combina o funcionamento de redes P2P com estruturas hierárquicas. Quando um ponto na rede deseja saber o endereço de um determinado *site*, ele envia uma requisição para o servidor de nomes mais próximo; se o mesmo não conhecer a tradução daquele nome a requisição é reenviada seguindo uma ordem hierárquica na escala de servidores de nomes. Quando a informação é obtida, ela "desce" na estrutura de servidores até chegar ao ponto que desejava a informação na rede para acessar o *site*. No caminho de volta ao ponto original, a tradução do endereço é armazenada nos servidores, otimizando assim as próximas consultas.

O sistema DNS e o Usenet são apenas dois exemplos clássicos do uso de estruturas P2P na formação e desenvolvimento da Internet, nos lembrando de seu aspecto histórico, evidenciando que o que foi novidade e surpresa em 2000 fazia parte das concepções originais dos criadores e primeiros desenvolvedores de sistemas para a Internet.

É, portanto, nesse cenário que podemos entender o P2P como alternativa tecnológica para a construção de sistemas colaborativos de ensino que não necessitem da mediação de um "ponto de encontro central" no papel de um servidor dos recursos da rede. E daí, podemos entender que o "*peer-to-peer* é uma classe de aplicativos que tira proveito de recursos – armazenamento, ciclos, conteúdo, presença humana – disponíveis nas margens da Internet. Tendo em vista que acessar esses recursos descentralizados significa operar em um ambiente de conectividade instável e endereços IP imprevisíveis, os nós P2P devem operar fora do DNS e ter uma autonomia total ou significativa a partir de servidores centrais" (Oram, 2001, pág. 24).

Vale aqui destacar esse aspecto de tirar proveito dos recursos

disponíveis nas margens da rede, ou seja, dos recursos periféricos da rede. É nessa periferia topológica que se incluem a quase totalidade dos usuários da Internet: usuários caseiros, redes de acesso público a informação, como Telecentros (programa de Inclusão Digital da Prefeitura Municipal de São Paulo), Infocentros (programa de Inclusão Digital do Governo do Estado de São Paulo), EICs (Escolas de Informática Comunitária, criadas pelo Comitê de Democratização da Informação), entre outros. São esses usuários, em contato com suas comunidades locais, que produzem uma grande quantidade de informações a cada dia; seja em forma de relatos, notícias, experimentações midiáticas, construção de novas soluções e outras possibilidades. Considerar uma estrutura técnica de desenvolvimento de redes de colaboração que possa levar em conta como premissa técnica essa capacidade de se relacionar com as bordas da rede é potencializar e criar mecanismos de conexão entre os mais diversos elementos dessa periferia, permitindo assim, o surgimento de conexões entre pessoas e de troca de informação das mais diversas possíveis.

Não menos importante, mas ainda relacionado com a questão do aproveitamento dos recursos "disponíveis nas margens da Internet" é a questão do poder de processamento técnico disponível e subutilizado no cenário cliente-servidor, onde a maior parte do processamento das aplicações utilizadas e dos serviços de acesso à informação é realizado pelos servidores. Apenas como efeito ilustrativo, podemos considerar os dados fornecidos por Oram (2001), onde se supõe que tenhamos em torno de cem milhões de PCs conectados à rede entre os trezentos milhões de usuários da Internet e que cada máquina conectada possua um poder médio de processamento de 100MHz e uma unidade de disco de 100MB. Levando-se esses dados em consideração, poderíamos observar que há um poder de processamento nas periferias da rede da ordem de dez bilhões de megahertz e dez mil terabytes de espaço em disco para armazenamento de informação e troca de dados.

Estabelecido o recorte técnico de estudo e centrada a questão da topologia P2P como alternativa a construção de sistemas colaborativos, podemos agora analisar algumas questões relacionadas ao contexto de uso da

tecnologia e seus desdobramentos.

Toda mudança ou ruptura estrutural de uma forma de relacionamento através da técnica traz como conseqüência a possibilidade de exploração e experimentação baseadas na nova técnica. Dado que os ambientes colaborativos de ensino baseados na tecnologia computacional são fortemente influenciados pelas premissas técnicas adotadas, temos que os aspectos cognitivos relacionados à troca de informação e a conseqüente validação da mesma como forma de construção do conhecimento torna-se também fortemente enquadrada numa visão oriunda da técnica e de suas topologias de relacionamento, de tal forma que "o estado das técnicas influi efetivamente sobre a topologia da megarrede cognitiva, sobre o tipo de operações que nela são executadas, os modos de associação que nela se desdobram, as velocidades de transformação e de circulação das representações que dão o ritmo a sua perpétua metamorfose" (Levy, 1993).

O que está em jogo nessa constatação de Levy é a maneira como se dá a circulação da informação e a forma como essa circulação permite a apropriação da informação pelo indivíduo, sua conseqüente transformação e reintegração à rede. Sabendo-se que a rede P2P tem como um de seus objetivos a criação de espaços que possam aproveitar os recursos que se encontram nas margens da Internet e que a topologia da rede não se vale necessariamente da mediação de um servidor para que possamos conectar nossas máquinas diretamente com outros computadores de usuários que também se encontram nas margens da rede, começamos a notar que "a topologia da megarrede cognitiva" foi alterada. Daí vem a afirmação de que "essa é uma violação completa do modelo de dados na Web, 'Conteúdo no centro', e o sucesso do Napster em violá-lo poderia ser chamado de 'Conteúdo nas margens'" (Oram, 2001, pág.31). O P de P2P passa a ser de, efetivamente, par , permitindo dessa forma que uma pessoa encontre um par e, como conseqüência desse encontro, os recursos que elas disponibilizam para o uso comunitário pela rede.

Sabendo-se que numa rede P2P uma máquina que deseja disponibilizar seus recursos deva estar conectada em um dado momento na rede para que a mesma possa ser acessada e que, muito provavelmente, o usuário se encontra

defronte do sistema nesses momentos, ocorre aqui o rompimento efetivo do paradigma da estrutura cliente-servidor. Numa rede P2P, o usuário encontra outro usuário e não apenas os recursos que foram um dia disponibilizados por ele em algum servidor do centro da rede. A mudança ocasionada pela relação conteúdo-pessoa proporciona uma série de possibilidades pedagógicas a serem exploradas pelos sistemas P2P que vierem a ser desenvolvidos com o foco da colaboração entre pares. Essas possibilidades pedagógicas são certamente marcadas pela modificação técnica, que é "*ipso facto* uma modificação da coletividade cognitiva, implicando novas analogias e classificações, novos mundos práticos, sociais e cognitivos." (Levy, 1993).

Basicamente, estamos falando que as redes de colaboração P2P formam o que Levy chama de tecnologias da inteligência, permitindo o compartilhamento e o armazenamento da informação de tal forma que essa rede em questão possa ser vista como um elemento de inteligência coletiva, não apenas um fórum de discussão e encontro, mas um mecanismo tecnológico que permite o desenvolvimento do pensamento, uma espécie de interface mediada pela tecnologia como uma nova forma de articular o raciocínio, desenvolver a articulação entre elementos icônicos que levam a potencialização de redes cognitivas que acabam por adquirir elementos emergentes, permitindo uma nova manipulação e decodificação dos fluxos de informação e o desenvolvimento de uma espécie de memória a longo prazo. Pode-se entender essa memória como sendo um pedaço do ciberespaço onde "a informação não se encontraria empilhada ao acaso, mas sim estruturada em redes associativas e esquemas. Estes esquemas seriam como 'fichas mentais' sobre as situações, os objetos e os conceitos que nos são úteis no cotidiano" (Levy, 1993).

Falar em elementos emergentes é falar de pequenas estruturas de dimensão local ao contexto no qual estão inseridas e que podem gerar um macrocomportamento observável. Segundo Johnson (2003), a emergência é composta pelo movimento das regras de baixo nível para a sofisticação do nível mais alto. As redes P2P como vemos nesse trabalho são redes que têm por objetivo o estímulo e desenvolvimento da colaboração em grupos locais. No entanto, o padrão de utilização da tecnologia por esses grupos locais começa a formar uma tendência emergente, ou seja, a colaboração *online*

estimulada pelos dispositivos técnicos locais estimula um macrocomportamento que poderia ser observado no momento em que se inicia o cruzamento das redes locais de colaboração, no momento em que pares de um determinado grupo local iniciam a troca de informações com outras redes de colaboração e tornam-se verdadeiros *hubs* – pontos de conexão (Barabasi, 2002) entre redes, permitindo a ampliação e efetivação desse macrocomportamento, que em hipótese alguma pode ser previsto ou antecipado por algum mecanismo regulador central.

Mas, também segundo Johnson (2003), "a complexidade sem adaptação é como os intricados cristais formados por um floco de neve: são bonitos, mas não têm função." Portanto, a mera conexão entre diversos grupos locais ou mesmo a conexão entre diversos pares numa rede só tem sentido quando se busca o compartilhamento de uma realidade comum, seja como objeto de estudo, seja como fórum de discussão para alternativas a problemas classicamente estabelecidos, seja como fonte de pesquisa de uma determinada mídia. É essa adaptação ao contexto, às inúmeras variáveis que fazem a mediação entre a colaboração efetiva e a mera conexão entre pares que se torna a referência do trabalho pedagógico em se tratando da colaboração em redes P2P.

Portanto, é nesse cenário técnológico e social que se dá o recorte técnico necessário para a realização desse trabalho e a contextualização das redes de colaboração P2P em relação as possibilidades de implementação da área. Como experimentação das idéias aqui pesquisadas e desenvolvidas, iremos implementar um aplicativo que forneça uma estrutura base para o suporte à colaboração em redes P2P.

A idéia por trás da construção desse aplicativo é implementar um ambiente de colaboração em rede que possa levar em consideração as características essencialmente técnicas do P2P como fator motivador para o desenvolvimento do trabalho colaborativo. No entanto, falar em trabalho colaborativo exige que busquemos uma dimensão dessa colaboração e o que esperamos atingir colaborativamente.

Para tanto, esse desenvolvimento tem seu foco centrado num ambiente formado por um grupo de estudantes, trabalhadores ou pesquisadores

trabalhando em conjunto com alguns interesses em comum, trabalhando em projetos comuns e atuando em atividades que podem ser desenvolvidas de forma isolada ou colaborativamente. O aplicativo desenvolvido tem por objetivo fornecer um suporte à colaboração dentro desse cenário, onde as pessoas podem tomar contato com a produção de um par na rede num dado momento do trabalho e até mesmo acompanhar o desenvolvimento de versões, a evolução do trabalho; podendo, em determinadas circunstâncias interferir nesse trabalho, propor novas diretrizes.

Esse modelo proposto passa pela compreensão e pela mediação de todas as questões relacionadas às possibilidades de uso e apropriação possibilitadas pela tecnologia P2P, desde a construção de uma estrutura de colaboração descentralizada e não mediada pelo papel técnico de um servidor, até a ocupação da periferia da rede com a codificação de conteúdo e o compartilhamento da informação de forma colaborativa.

Portanto, o objetivo do estudo realizado nesse trabalho e da implementação desenvolvida vem contemplar as questões acima descritas e oferecer uma experiência de desenvolvimento como fator de análise e crítica da tecnologia e seus desdobramentos, trazendo como contriuição na área um ambiente experimental em P2P.

Este trabalho apresenta os resultados dos estudos desenvolvidos e para tanto está estruturado da seguinte forma: no Capítulo 2 temos uma descrição das tecnologias e especificações pesquisadas como forma de construir uma fundamentação teórica para o desenvolvimento do trabalho. O Capítulo 3 apresenta uma descrição da proposta de trabalho e seus respectivos módulos de funcionamento. O Capítulo 4 traz uma descrição da implementação realizada e uma avaliação do trabalho com base nos conceitos explorados no Capítulo 2. O Capítulo 5 faz uma análise comparativa de uma aplicação cliente/servidor em relação a solução apresentada no capítulo 4 e apresenta uma proposta híbrida de solução cliente/servidor e P2P. O Capítulo 6 apresenta a conclusão do trabalho e sugestões de trabalhos futuros.

Capítulo 2 – Fundamentação Teórica

Ao iniciar o estudo e o desenvolvimento de um projeto de sistema colaborativo em P2P, torna-se necessário compreender essencialmente três pontos: o que entendemos como sendo uma rede P2P, quais as características de sistema necessárias para a construção efetiva de um ambiente remoto de colaboração e a dinâmica de interação nesse tipo de rede, o que significa descrever parte do processo social vivenciado pelo encontro com o outro através da mídia formada pela interface do sistema e pela malha de possibilidades oferecida pela topologia de rede.

Como definição de P2P, podemos nos valer do conceito utilizado por Schoder e Fischbach (2003), onde "P2P refere-se a toda tecnologia que permite que dois ou mais pares possam colaborar espontaneamente numa rede de iguais (pares) usando as informações apropriadas e sistemas de comunicação sem necessidade de alguma forma de coordenação central". Portanto, é essa característica de uma estrutura de comunicação que permita a troca de informações de forma a não se valer de uma estrutura centralizada que se pretende contemplar.

Pensar em um sistema de colaboração, seja ele baseado em P2P ou na estrutura cliente-servidor, nos leva necessariamente a buscar compreender quais são as características de sistema que permitem que essa colaboração possa ocorrer efetivamente. Segundo o trabalho de Ionescu *et al.* (1999), qualquer tipo de ambiente de colaboração remota deve ser composto por três componentes, essencialmente:

- deve possuir um mecanismo de sincronização que mantenha todas as aplicações remotas com as mesmas características;
- deve conter um mecanismo de colaboração ou um protocolo que possa gerenciar as demandas de comunicação das aplicações de colaboração remota. Vale descatar aqui o entendemos neste trabalho pelo conceito de colaboração e, mais especificamente, colaboração em rede. A colaboração consiste em si da atividade de prestar uma determinada assistência, auxílio na realização de uma determinada

tarefa. Já a colaboração em rede consiste nesse auxílio através dos fluxos de conversação, ou seja, dos fluxos que permitem a troca de informações entre pares numa rede.

- deve possuir um sistema de comunicação que possa suportar os dois componentes acima citados de forma a garantir a escalabilidade e gerenciamento das sessões de colaboração.

Foi levando-se em consideração as características acima citadas que buscamos nortear a pesquisa de protocolos e estruturas de comunicação em P2P que permitissem contemplar esses três componentes.

Em relação ao processo social, podemos dizer que ele tem início no momento em que compreendemos que a topologia da rede P2P torna necessário que, na busca de um determinado recurso, encontremos um determinado par na rede, que não é apenas uma máquina, mas sim uma pessoa que disponibiliza material de seu interesse, tanto da autoria de terceiros como de produção própria, e está conectada na rede no momento de nossa busca. Está estabelecido o encontro. Potencialmente, temos acesso não apenas ao recurso, mas a quem o disponibilizou. Podemos, de uma forma ou de outra, estabelecer um vínculo com esse par na rede, realimentar sua produção, indicar assuntos correlatos, enfim, derivar desse vínculo um processo efetivo de interação par-a-par. Conseqüentemente, podemos dizer que "as tecnologias *peer-to-peer* retornam a Internet à sua visão original: todo mundo cria, todo mundo usa" (Oram, 2001). É dessa experiência que podemos notar que encontrar o outro produtor ou replicador de um determinado recurso é sensivelmente diferente da experiência de encontrarmos o recurso deslocado de seu ponto original de inserção na rede. Surge daí a idéia de que a periferia (os pontos periféricos em uma topologia cliente/servidor clássica) passa a ser o centro da rede.

Sabemos que "elaborar uma proposição ou uma imagem é, portanto, o mesmo que construir vias de acesso a essa representação na rede associativa da memória de longo prazo" (Levy, 1993). É o reflexo dessa rede associativa que a tecnologia P2P permite que encontremos nos espaços públicos em disco disponibilizados pelos pares que compõem uma determinada rede. Essa rede, portanto, complementa-se com nossas próprias formas de representação dos

recursos que pretendemos disponibilizar e, se pararmos para prestar atenção em nossos vizinhos na rede, poderemos notar que "informação local pode levar à sabedoria global" (Johnson, 2003). É do cruzamento dos mais diversos nós da rede que se dá efetivamente a dinâmica de interação P2P. Para se ter uma visão mais pragmática da dinâmica de interação, iremos analisar cinco sistemas de colaboração (iKnow, Edutella, LionShare, Jabber e ICQ) baseados na tecnologia P2P.

Essa dinâmica de interação tem levado a um uso crescente de sistemas P2P nos últimos tempos, mais diretamente os sistemas de troca de arquivos, tais como o Napster, Freenet, Kazaa, Morpheus, entre outros. Esse interesse crescente traz como conseqüência uma reflexão sobre o padrão de funcionamento dessas redes e de suas possibilidades tecnológicas e sociais derivadas da experiência de interação adquiridas até o presente momento. Um fato que podemos evidenciar em meio a essa reflexão é a questão da escalabilidade desse tipo de rede. Como se dá o crescimento dessa rede? Quais seriam os pontos de controle e possíveis pontos de falha do sistema? O P2P é realmente uma estrutura desprovida de hierarquia ou apenas uma maior descentralização em relação aos sistemas cliente-servidor atualmente em uso? A resposta a essas perguntas nem sempre é intuitiva, até porque algumas definições na área ainda são nebulosas e arduamente discutidas pelos grupos envolvidos com o desenvolvimento dessas redes. No entanto, parte do estudo que aqui apresentamos busca compreender os aspectos gerais do funcionamento de indexação e busca em sistemas P2P atualmente em funcionamento. O foco do trabalho visa possibilitar um ambiente livre e não-hierárquico de troca de conteúdo entre aprendizes. Por conteúdo, podemos entender como sendo qualquer estrutura lógica computacional que sirva como subsídio de expressão para uma determinada mídia.

Essa troca de conteúdo, no contexto de uma rede P2P, perde a mediação, que poderia ser tanto de conteúdo como de acesso, que era antes desempenhada pelo provedor do serviço no papel técnico do servidor da informação. Essa lógica é invertida e desaparece o papel centralizador dessa entidade de mediação, dado que a troca de conteúdo passa a ocorrer num mecanismo tecnológico que propicia o encontro entre duas ou mais

pessoas. Sendo assim, entendemos que desde o processo de busca, do encontro e da aquisição da informação, o fluxo de dados percorre máquinas que podem ser identificadas por um determinado usuário conectado no instante da interação, que é membro de uma determinada comunidade de pares. O caminho traçado na rede permite então colocar em contato quem produziu ou replicou determinado recurso e quem o deseja. É nessa possibilidade de interação que existe a brecha para a construção de novos paradigmas que possam aproximar cada vez mais a tênue barreira entre o real e o virtual.

Além das questões acima tratadas, aparece no freqüentemente nos sistemas de colaboração em rede a questão da reputação dos pares do sistema. Dessa forma, iremos neste capítulo de fundamentação teórica discutir essa questão e apresentar algumas soluções pesquisadas.

Para visualizar tecnicamente esse contato entre pares e a dinâmica de funcionamento dos três componentes essenciais para um sistema de colaboração remota, alguns protocolos de interação e construção de redes são estudados, bem como suas implementações.

2.1 Protocolos P2P

Conforme as pesquisas realizadas, chegamos a um conjunto de protocolos P2P que são as referâncias na área: Chord (Stoica *et al.*, 2001), CAN (Ratnasamy *et al.*, 2001), Pastry (Rowstron e Druschel, 2001), Tapestry (Zhao *et al.*, 2001), Gnutella (http://www.clip2.com) e JXTA (Traversat *et al.*, 2003). Com base nos estudos dos trabalhos relacionados a cada um desses protocolos, pudemos destacar alguns pontos críticos comums a cada um dos trabalhos. Ao dar destaque a esses pontos, podemos ter uma visão conceitual de cada uma das propostas de trabalho e compará-las entre si. Os pontos a serem analisados em cada um dos protocolos são a inserção de um novo nó na rede, a exclusão de um nó da rede, distribuição/localização de recursos, roteamento de mensagens e tolerância a falhas/redundância.

2.1.1 Chord

O protocolo Chord trabalha com a idéia de uma de roteamento distribuída que é implementada com base numa tabela *hash* distribuída.

Segundo Ricarte (www.dca.fee.unicamp.br/cursos/PooJava/collections/h_hasht.html), uma tabela *hash* é uma estrutura que permite associar uma chave a um valor e, posteriormente, ter acesso ao valor a partir de sua chave associada. A chave é transformada por uma função em uma posição na tabela; usando sempre essa transformação, a localização de chaves na tabela é realizada rapidamente. Cada nó da rede somente mantém informações a respeito de O(log N) nós, onde N é o número total de nós na rede, sendo possível realizar qualquer busca de informações com base nessa tabela de roteamento.

Chord utiliza uma função *hash* consistente que associa a cada chave e nó um identificador de m-bits, usando para isso uma função do tipo SHA-1. O identificador do nó é escolhido com base em seu endereço IP (*Internet Protocol*) e o identificador da chave é escolhido com base no valor da chave, que pode ser um nome de arquivo, nome de imagem, uma URL (*Uniform Resource Locator*), enfim, um valor que identifique o recurso que se deseja compartilhar na rede.

2.1.1.1 Distribuição/localização de recursos

As chaves são associadas aos nós da rede através da formação de um círculo de identificação. Uma chave K é associada ao primeiro nó cujo identificador seja igual ou siga o valor do identificador de K no espaço de identificação. A esse nó, chamamos de *nó sucessor* de K.

2.1.1.2 Inserção de um novo nó na rede

Duas premissas são levadas em consideração na inserção de um novo nó na rede: cada nó sucessor é corretamente mantido e para cada chave K seu nó sucessor responsável permanecerá inalterado. Cada nó numa rede Chord possue um ponteiro para seu nó predecessor, que contém o identificador Chord e o endereço IP do predecessor imediato do nó em questão. Quando um nó é inserido na rede, seu predecessor e sua tabela de roteamento são inicializadas com base nos valores de um nó

conhecido; os nós existentes atualizam seus ponteiros predecessores e suas tabelas de roteamento para identificar o novo nó e há uma notificação para a camada superior de programa, que desta forma irá transferir para o novo nó os valores de identificador de chave que daqui por diante ele será responsável.
A maneira como um novo nó deve conhecer algum nó pré-existente para ser inserido na rede não é relatada no artigo de referência do protocolo.

2.1.1.3 Exclusão de nó da rede
É feita da mesma forma que a inclusão, ficando a camada de programa responsável pela redistribuição de chaves anteriormente armazenadas no nó excluído.

2.1.1.4 Roteamento de mensagens
É feito com base nas tabelas de roteamento de cada nó, que direcionam uma mensagem com base no identificador de chave armazenado nas mesmas.

2.1.1.5 Tolerância a falhas/redundância
Quando um nó n sai da rede inesperadamente, os nós que o registraram em sua tabela de roteamento precisam agora encontrar um sucessor de n. Para auxiliar nessa tarefa, o protocolo Chord mantém uma lista de com alguns sucessores. Essa tabela de próximos sucessores também auxilia na questão de replicação dos identificadores de chave na rede, evitando assim a existência de pontos únicos de falha para determinados conjuntos de identificadores. Uma aplicação típica utilizando Chord irá enviar réplicas dos dados associados a uma chave a Z nós que sucedem a chave, onde Z é o número que nós que podemos estabelecer como pontos de redundância para os dados.

2.1.2 CAN

O protocolo CAN trabalha com a idéia de que cada nó da rede armazena informações a respeito de uma *zona* da tabela *hash* completa. Um nó apenas tem informações a respeito de um pequeno número de zonas adjacentes a que ele se encontra. Essas zonas podem ser compreendidas como integrantes de um plano cartesiano virtual de d-dimensões. Em um momento qualquer, podemos notar que todas as coordenadas do espaço virtual são dinamicamente particionadas entre todos os nós do sistema, de tal forma que cada nó possua uma zona própria e distinta das demais dentro de todo o espaço.

2.1.2.1 Distribuição/localização de recursos

Para armazenar o par de dados <chave,valor>, a chave é deterministicamente mapeada para um ponto P no espaço através de uma função *hash* uniforme, que é uma função que mapeia cada chave em um inteiro no intervalo (0, N-1), onde N é a capacidade do arranjo das células para os pares <chave,valor>, e o par de dados é então armazenado no nó da rede que ocupa o espaço onde o ponto P está inserido.

2.1.2.2 Inserção de um novo nó na rede

Assim que um novo nó entra na rede, ele precisa alocar para si um pedaço do espaço de coordenadas. Isto é feito quando um nó existente divide sua própria zona alocada pela metade, cedendo uma metade para o novo nó. Os vizinhos dessa nova zona são informados da presença desse nó e, assim, podem atualizar sua tabela de roteamento. No entanto, uma questão que se torna importante na análise de todos os protocolos é a forma como um novo nó descobre o endereço IP de um nó já presente na rede. Esse fator também não é resolvido pelo protocolo CAN, sendo que é apenas proposta uma estrutura clássica de DNS, onde um nó poderia se conectar e obter um IP válido para acesso a rede e obter uma tabela de roteamento inicial.

2.1.2.3 Exclusão de nó da rede

Quando um nó parte da rede, deve-se levar em consideração a questão de que sua zona deve ser ocupada pelos nós remanescentes. Há duas opções nesse caso. Se a zona livre pode ser agrupada com a zona de um nó vizinho e produzir uma única zona válida, então esse procedimento é adotado. Se isso não for possível, então a zona livre é entregue ao vizinho que possui a menor zona e esse nó vai lidar temporariamente com a junção das duas.

2.1.2.4 Roteamento de mensagens

O roteamento numa rede CAN se dá pelo cálculo do caminho direto no espaço cartesiano entre o ponto de origem e o ponto de destino, através das coordenadas do espaço. Uma mensagem CAN inclui em seu escopo as coordenadas do ponto de destino. Há vários caminhos de um ponto a outro; portanto, se há problemas em um nó, o protocolo facilmente redireciona a mensagem para outro caminho.

2.1.2.5 Tolerância a falhas/redundância

Se um nó da rede cai repentinamente, o protocolo CAN provê um algoritmo que permite a um vizinho desse nó tomar posse da zona. No entanto, os pares de informação <chave,valor> ficarão indisponíveis até os detentores dos dados enviarem renovações dos dados, o que é sempre feito regularmente como forma de manutenção da rede.

2.1.3 Pastry

Um nó numa rede mantida pelo protocolo Pastry mantém o registro de seu vizinho imediato no espaço de identificação dos nós, notificando o nível de programa da chegada de um novo nó, da partida de um nó ou da falha de um nó. Ele leva em consideração a localidade geográfica física dos nós na rede para distribuí-los como vizinhos, o que não ocorre nos outros

protocolos, buscando dessa forma minimizar a distância a ser percorrida por qualquer mensagem que for transitar na rede.

2.1.3.1 Distribuição/localização de recursos

Cada nó na rede Pastry possui um identificador individual de 128 bits. Esse identificador serve para indicar a posição de um nó num espaço circular de identificação. O identificador dos nós é gerado através de funções *hash* criptografadas computando-se o identificador público de um nó ou o número IP do nó. As chaves geradas em relação aos conteúdos inseridos na rede são valores que se comparam com o identificadores de nós, permitindo-se, assim, a inclusão das chaves nos pontos da rede relacionados.

2.1.3.2 Inserção de um novo nó na rede

Quando um nó chega na rede, ele irá inicializar suas tabelas de roteamento e informar os outros nós da sua presença. Assume-se que esse nó conheça um participante da rede Pastry de início. A obtenção da informação a respeito desse nó poderia se dar através da informação de um administrador de sistema ou através de uma requisição enviada em *multicast* IP num anel em expansão de máquinas conhecidas pelo sistema.

Supondo ser o novo nó X na rede, ele irá questionar o nó conhecido previamente, digamos A, para enviar uma mensagem de *join*, que é a mensagem do novo par informando de sua entrada na rede, com a chave igual a X. Pastry irá rotear essa mensagem para um nó existente, digamos Z, cujo ID seja numericamente próximo a X, sendo que no meio do caminho de A até Z, todos os nós irão atualizar suas tabelas de roteamento para incluir o novo nó X.

2.1.3.3 Exclusão de nó da rede

Quando da partida de um nó da rede, seu vizinho no espaço de

identificação contacta o nó com maior índice ao lado do nó excluído e o questiona por suas informações de roteamento, substituindo-o dessa forma.

2.1.3.4 Roteamento de mensagens

Pastry irá rotear uma mensagem para o nó cujo ID for numericamente próximo a uma dada chave. Em cada passo do roteamento, um nó normalmente reenvia a mensagem para um nó cujo ID divida com a chave um prefixo que é pelo menos um dígito (ou b bits) maior que o prefixo que a chave divide com o presente nó, onde a mensagem atualmente se encontra. Se nenhum nó que atinge essa condição é conhecido na tabela de roteamento, a mensagem é enviada para um nó cujo ID divida o mesmo número de dígitos com a chave que o nó atual, mas que seja numericamente mais próximo da chave.

2.1.3.5 Tolerância a falhas/redundância

Através de um mecanismo de notificação, o protocolo Pastry permite uma aplicação manter réplicas de arquivos em K nós próximos ao nó que armazena a chave, evitando assim gargalos de conteúdo, pontos únicos de acesso a determinado conteúdo, no caso de falha de um nó ou partida de um nó da rede.

2.1.4 Tapestry

O protocolo Tapestry tem suas raízes na técnica de busca distribuída Plaxton (busca de objetos distribuídos numa rede que satisfaz a requisição de objetos criando uma cópia do mesmo nas "proximidades" do nó que o requisitou). No entanto, ele foi melhorado nos quesitos escalabilidade e adaptação em relação a possíveis falhas e ataques no ambiente de rede. Cada máquina neste protocolo pode assumir o papel de servidor (onde objetos são armazenados), roteador (transmitindo mensagens) e cliente (originando requisições).

2.1.4.1 Distribuição/localização de recursos

Os objetos e os nós possuem nomes independentes de sua localização e propriedades semânticas. Esses nomes são gerados por seqüências aleatórias de bits fixas representadas por uma base comum (160 bits, por exemplo). As chaves são distribuídas entre os nós e objetos por uma função *hash*, tal como SHA-1 (*Secure Hash Algorithm*). Um servidor S publica que ele tem um objeto O roteando uma mensagem para o "nó raiz" (nó que contém o recurso) de O. Em cada nó do caminho de S até o nó raiz pelo qual a mensagem passa, fica armazenado informações que irão montar um mapa no formato <Object-ID(O),Server-ID(S)>. É através desse mapa que os recursos poderão ser localizados na rede. O nó raiz é um ponto crítico na implementação desse protocolo. Se ele ficar fora da rede, o recurso torna-se indisponível. Há iniciativas para resolver esse problema em implementação, como, por exemplo, associar a mais de um nó raiz um determinado recurso. O protocolo toma o ID do objeto a ser procurado gerado pela função de *hash* e busca uma série de nós raízes em potencial para realizar a busca pelo recurso.

2.1.4.2 Inserção de um novo nó na rede

Assume-se que um nó N conhece um *gateway* G que é um nó na rede Tapestry. Inicialmente, o nó N toma o mapa de roteamento de G como ponto de partida.

2.1.4.3 Exclusão de nó da rede

O nó passa a não mais fazer parte das mensagens de atualização das tabelas de roteamento e, portanto, passa a ser ignorado pelo restante da rede. Seu conteúdo permanecerá em *cache* temporariamente e depois será apagado.

2.1.4.4 Roteamento de mensagens

O protocolo Tapestry, através do algoritmo Plaxton, usa mapas

de roteamento local, que são chamados de "mapas dos vizinhos" que incrementam as mensagens de roteamento até o ID de destino, dígito por dígito. Um nó N tem um mapa de vizinhos com múltiplos níveis, onde cada nível representa um sufixo a mais na posição de dígitos do ID a ser encontrado. Por definição, o enésimo nó que uma mensagem atinge compartilha um sufixo de pelo menos um tamanho n com o ID de destino. Para encontrar o próximo roteador, devemos olhar para o nível n+1 no mapa e procurar entradas que combinem o valor do próximo dígito no ID de destino.

2.1.4.5 Tolerância a falhas/redundância

O protocolo Tapestry implementa um mecanismo em nível de programa que provê uma cópia em *cache* do conteúdo disponibilizado por um nó. Esse *cache* é constantemente atualizado por mensagens periódicas de atualização ou seu conteúdo removido no caso da falta dessas mensagens. Dessa forma, o protocolo lida com falhas de forma natural e não como um caso de exceção em seu funcionamento. Para detectar falhas de *links* e servidores, o protocolo utiliza o tempo limite do protocolo TCP. Fora isso, o protocolo utiliza mensagens periódicas via UDP para os vizinhos verificando sua existência.

2.1.5 JXTA

O projeto JXTA (*Juxtapose*) teve origem na Sun Microsystems e hoje é um projeto de código livre mantido com apoio da Sun. A idéia do projeto é criar uma camada de rede virtual no topo da infra-estrutura de rede física existente com base numa série de especificações, os padrões dos protocolos que compõem o JXTA, que pudessem servir como referência para a construção de redes P2P. Atualmente, o projeto conta com implementações do protocolo em linguagem Java e C. Além disso, o JXTA é um sistema aberto e busca uma estrutura tecnológica de interoperacionalidade devido aos

inúmeros projetos desenvolvidos pela comunidade. Devido a essas características e outros argumentos técnicos que serão analisados ao longo do texto, o adotamos como protocolo base para o desenvolvimento da aplicação das idéias desenvolvidas nesse trabalho.

A versão 2.0 das especificações de protocolos do JXTA introduzem o conceito do *rendezvous-peer view* (RPV) para a propagação de mensagens de resolução de endereços e um índice distribuído de compartilhamento de recursos (SRDI, *Shared Resource Distributed Index*) que indexa mensagens no foco de visão do par *rendezvous* para que a consulta a mensagens se torne mais eficiente.

A implementação do projeto JXTA é baseada em cinco abstrações para a construção da rede virtual imaginada que nos permitem compreender sua dinâmica de funcionamento. Primeiro, há um modelo de endereçamento lógico por par na rede que se espalha por toda a rede JXTA. Segundo, grupos de pares na rede podem se organizar em *peergroups*, que nada mais são do que uma espécie de domínio virtual na rede. Terceiro, um sistema de mensagens (*advertisements*) para publicar os recursos armazenados por um par na rede. Quarto, um mecanismo universal de agregação, chamado de *resolver,* que realiza todas as operações de agregação necessárias num sistema distribuído. Por último, um sistema de canais de comunicação virtual chamado *pipe* que permite que as aplicações possam se comunicar entre elas.

O modelo de endereçamento do JXTA é um modelo lógico uniforme e independente da localização do par na rede. Para cada recurso existente na rede, seja um par, um canal de comunicação virtual (*pipe*), um dado qualquer ou um grupo de pares, há um ID único relacionado de 128 bits aleatórios UUIDs (*Universal Unique Identifier*). Esse recurso permite uma característica de fundamental importância para o bom funcionamento da rede P2P, que é a questão do endereçamento único independente do valor IP de uma máquina para pares na rede. Por exemplo, supondo-se uma situação em que uma máquina se conecte na rede via DHCP (*Dynamic Host Configuration Protocol*, protocolo de atribuição dinâmica de endereços IP a máquinas numa rede), o que acontece com freqüência quando usamos acesso a Internet em provedores públicos com acesso discado via modem, a máquina terá um endereço IP para

cada sessão e se seu endereçamento na rede P2P fosse baseado no número IP, teríamos sérios problemas de conectividade. No entanto, com o endereçamento de 128 bits UUIDs a máquina tem seu endereço próprio para a rede P2P, independente de seu número IP em uma determinada sessão de acesso a Internet, permitindo que ela possa ser acessada continuamente quando estiver conectada em várias sessões de uso diferentes.

As mensagens (*advertisements*) na rede JXTA são basicamente estruturas de metadados que descrevem os recursos que representam baseadas em XML. Os pares na rede armazenam, publicam e trocam mensagens para descobrir recursos disponibilizados por outros pares na rede. Todas as mensagens têm um tempo de vida no qual circulam pela rede, o que permite que as mensagens sejam excluídas sem a necessidade de uma coordenação central.

Vale a pena destacar alguns pontos importantes na abstração do projeto JXTA que merecem ter uma consideração à parte.

2.1.5.1 Super Pares Rendezvous

Os pares de Rendezvous têm uma condição especial na rede P2P JXTA, o que permite que os chamemos de super pares. Eles são responsáveis por armazenar os índices das mensagens que são enviadas pelos outros pares para a rede, ou seja, armazenam ponteiros para os pares que ficam na periferia da rede e que armazenam a mensagem propriamente. São pares conhecidos na rede pelo papel que desempenham.

É importante ressaltar aqui que qualquer par numa rede JXTA pode ser um par Rendezvous assumindo que os mesmos tenham as credenciais corretas para isso, o que pode ser decidido dentro do âmbito de um grupo de pares. Um dos motivos para que alguns pares na rede tenham essa característica especial é o fato da rede P2P ser muito dinâmica, ou seja, entram e saem pares o tempo inteiro. Fato este que torna computacionalmente custoso manter em um grupo de pares uma visão global da rede sem assumir algumas estruturas mais centrais na abstração da rede

que possam ter uma visão global de todo o sistema. Essa é também uma diferença importante do JXTA em relação a protocolos como Chord ou CAN, que são protocolos que não presumem uma rede dinâmica dessa forma, imaginando em sua concepção que a saída e entrada de pares na rede não é uma constante. Característica esta que certamente acarreta em perda de performance devido ao custo computacional da montagem constante e rearranjo das tabelas de roteamento. Dessa forma, cada par Rendezvous mantém sua própria imagem dos outros pares Rendezvous no grupo ao qual ele faz parte. Essa imagem nada mais é do que uma lista ordenada de todos os outros pares Rendezvous pelos seus respectivos Ids. Quando um par novo entra na rede, ele se conecta com algum par Rendezvous conhecido previamente, o que permite que ele adquira a tabela de todos os pares Rendezvous que aquele inicial enxerga, podendo se inserir na topologia da rede. Quando esse par ou quaquer outro na rede deseja obter alguma informação, ele envia uma requisição ao par Rendezvous mais próximo, que irá gerar com base em sua função *hash* um mapeamento para verificar em qual par Rendezvous se encontra o ponteiro para aquela informação desejada. Ao chegar a mensagem até o par Rendezvous correto, a requisição é então encaminhada ao par que detém a informação desejada.

Vale lembrar que cada par Rendezvous replica sua tabela de índices em pelo menos mais dois pares, um acima numericamente e um abaixo numericamente dele, de tal forma que se esse par sair da rede os ponteiros armazenados por ele não são perdidos do sistema. Essa característica é importante, pois ela permite que não haja tráfego na rede somente para a manutenção das tabelas de roteamento, economizando dessa forma banda para outras funções, o que não ocorre nos outros protocolos estudados. Supondo-se que haja uma requisição mapeada para um par Rendezvous que já não mais se encontra na

rede, é nesse caso que entra em ação um mecanismo chamado *walker*, que fará com que a busca seja executada em alguns pares Rendezvous com IDs numericamente acima e numericamente abaixo do par onde deveria estar armazenado o ponteiro para a informação desejada. O número de pares acima e abaixo que o mecanismo de *walker* pode consultar é configurado nos parâmetros do protocolo. Dessa forma, as possibilidades de encontrar um recurso, mesmo que a rede sofra alterações topológicas drásticas, ainda é considerável.

2.1.5.2 Super Pares Conexão

O projeto JXTA introduz um conceito em redes P2P dos Super Pares de Conexão, que nada mais são do que pares que permitem interconectar pares que não possuem ligação física direta, por exemplo, pares dentro de uma rede com *firewall* ou NAT (*Network Adress Translation*). Da mesma forma que os pares de Rendezvous, qualquer par na rede pode assumir a posição de um Super Par de Conexão desde que possua as credenciais suficientes, o que pode ser validado dentro do escopo de um grupo de pares.

É importante ressaltar o aspecto transiente das informações armazenadas por um Super Par de Conexão, pois ele irá armazenar as informações relativas aos pares da periferia da rede numa dada sessão de conexão. Isto pode não se repetir na próxima sessão, dado que o mesmo par periférico pode se conectar em outro Super Par de Conexão, desde que disponível.

2.1.5.3 Grupos de pares

Um grupo de pares representa um conjunto dinâmico de pares que possuem um determinado interesse em comum e estão de acordo com algumas regras comuns, como a política de inclusão de usuários, a troca de conteúdo, credenciais, etc.

Há três razões centrais para a criação de um grupo de pares e

certamente a questão da segurança é uma delas. Criar um grupo de pares permite evitar que pares não desejados façam parte da topologia dessa rede, mesmo que esses pares se encontrem na mesma topologia física de rede. A segunda razão é criar um escopo para o ambiente de rede, ou seja, reunir pares dentro de um conjunto de interesses comuns. A terceira razão é o monitoramento da rede e o acompanhamento do tráfego.

2.1.5.4 Canais de comunicação (Pipes)

Os *pipes* são canais de comunicação virtuais que servem para enviar e receber mensagens entre os diversos serviços e aplicações rodando na rede P2P. Esses canais trabalham com a abstração de um *pipe endpoint*, que representa um ponto de entrada e saída, abrindo e fechando por "completo" o canal de comunicação. Os canais podem se conectar em um ou mais *endpoints*, dando a possibilidade de se enviar mensagens em *multicast*. Cada *pipe* possui também sua ID única que o identifica,o *Pipe ID*.

Essa abstração permite dois modos de comunicação, o *pipe* ponto–a–ponto, que conecta exatamente dois pares na rede em um canal de comunicação unidirecional e assíncrono; e o *pipe* de propagação que conecta um *pipe* de saída com vários de entrada, permitindo efetivamente o recurso de envio de mensagens em *multicast*.

2.1.5.5 Conjunto de Protocolos

O projeto JXTA consta de seis protocolos em sua especificação, sendo que são divididos em duas categorias, que podem ser observadas na figura 2.1.

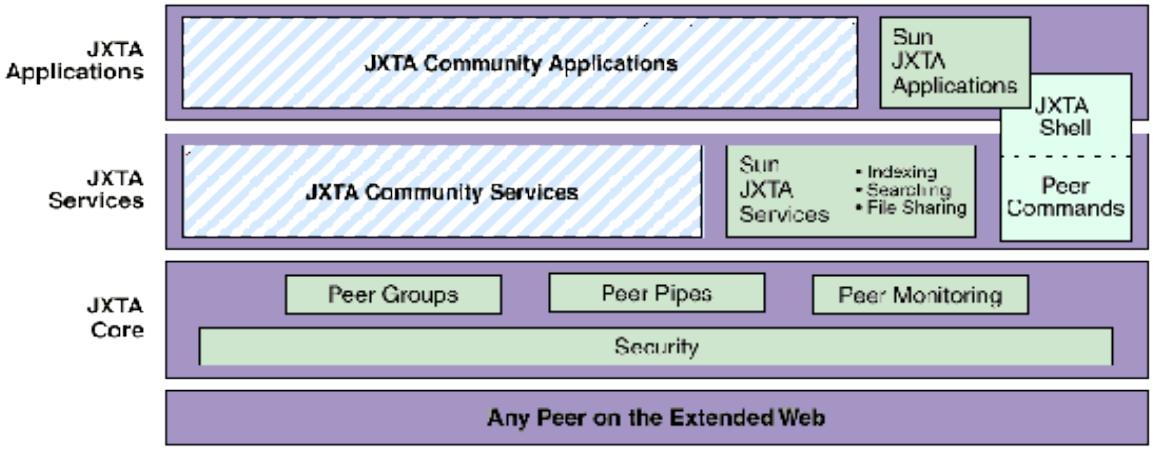


Figura 2.1 – Camadas de protocolos da Especificação JXTA. (Fonte: http://jxta.org)

Há protocolos de especificação do núcleo do sistema, que são os requisitos mínimos para o funcionamento de qualquer rede nos padrões JXTA, e protocolos de serviços básicos, que são os componentes e comportamentos necessários para as implementações básicas oferecidas pelo próprio projeto JXTA. No entanto, eles podem ser substituídos por soluções proprietárias se assim for conveniente para alguma determinada atividade. Na figura 2.1, a camada de aplicações representa os sistemas que possam vir a ser desenvolvidos utilizando as camadas de protocolos do JXTA.

Os protocolos de especificação do núcleo são dois. O ERP (*Endpoint Routing Protocol*) é um protocolo pelo qual o par na rede pode descobrir uma rota (sequência de passos na rede) e é usado para se enviar uma mensagem para outro par na rede. O PRP (*Peer Resolver Protocol*) é o protocolo pelo qual um par pode enviar uma mensagem de resolução de endereços para um ou mais pares e receber uma resposta ou várias respostas.

Os protocolos de serviços básicos são quatro. O RVP (*Rendezvous Protocol*) é o protocolo pelo qual os pares podem subescrever ou serem subescritos num serviço de propagação de

mensagens. O PDP (*Peer Discovery Protocol*) é o protocolo pelo qual um par na rede pode publicar suas próprias mensagens e descobrir mensagens de outros pares, que podem ser notificação de um novo par na rede, formação de um grupo de pares, constituição de *pipes* e conteúdos disponibilizados. O PIP (*Peer Information Protocol*) é o protocolo pelo qual um par pode obter informações de estado sobre outros pares, tais como o estado de ativo ou não, tempo de conexão na rede, carga de tráfego, capacidades, etc.. O PBP (*Pipe Binding Protocol*) é o protocolo pelo qual um par pode estabelecer um canal virtual de comunicação entre dois ou mais pares.

Agora, como forma de podermos analisar o protocolo JXTA em comparação com os protocolos anteriormente estudados, iremos destacar o funcionamento desse protocolo em relação aos seguinte pontos: a inserção de um novo nó na rede, a exclusão de nó da rede, distribuição/localização de recursos, roteamento de mensagens e tolerância a falhas/redundância.

2.1.5.6 Inserção de um novo nó na rede

Um novo nó se insere na rede JXTA através do contato com um Super Par de Conexão, de onde o novo par está diretamente inserido nas tabelas de roteamento e na topologia da rede P2P.

2.1.5.7 Exclusão de um nó da rede Quando um par sai da rede, o conteúdo por ele disponibilizado através das mensagens que foram por ele enviadas ao sistema tem um tempo de duração limitado. Isto permite que após decorrido certo intervalo de tempo essas mensagens, por não terem sido renovadas, percam sua validade, o que faz com que os registros (ponteiros) das mensagens enviadas pelo par que residiam no Super Par de Rendezvous sejam automaticamente excluídos sem a necessidade de uma

organização centralizada.

2.1.5.8 Distribuição/localização de recursos

A distribuição e localização é feita basicamente baseada na estrutura oferecida pelos Super Pares de Rendezvous, que são os responsáveis na rede por rotear as requisições de informação para os pares respectivos na periferia da rede. Para realizar esse trabalho, esses Super Pares armazenam ponteiros para cada publicação de mensagem pelos pares periféricos. Assim sendo, eles são consultados quando desejamos obter a informação de onde determinado recurso se encontra na rede.

2.1.5.9 Roteamento de mensagens

É feito com base no descobrimento de uma rota de pares a ser percorrida. O descobrimento da rota passa fundamentalmente pelos Super Pares de Conexão e Super Pares de Rendezvous, que são os responsáveis na rede por armazenar as rotas para pares periféricos. Se o ponteiro para uma determinada mensagem não se encontra no Super Par de Rendezvous esperado, entra em funcionamento o mecanismo do *walker*, que irá continuar a busca nos Super Pares numericamente superiores e inferiores ao ID do Super Par inicial.

2.1.5.10 Tolerância a falhas/redundância

Um sistema que previne a rede JXTA de pontos únicos de gargalo é a replicação do conteúdo dos Super Pares de Rendezvous em pares próximos numericamente, evitando assim que se um par sai da rede os ponteiros para os recursos armazenados nos pares periféricos sejam perdidos.

2.1.6 Gnutella

O Gnutella foi desenvolvido como um protocolo basicamente para busca de recursos distribuída num ambiente de rede. O protocolo pode tanto funcionar num paradigma cliente/servidor quanto *peer-to-peer.* No entanto as características que o distinguem são suas capacidades em redes P2P. Nesse modelo, as máquinas tanto são clientes quanto servidores e são chamadas de *servents*. As especificações do protocolo Gnutella são descritores da forma como os *servents* vão se comunicar entre si, ou seja, as regras de como essa comunicação deve ocorrer.

Para a efetiva comunicação, o protocolo propõe um conjunto de cinco descritores: *ping, pong, query, queryHit* e *push*. O *ping* é usado para descobrir os pontos presentes na rede; *pong* é a resposta que um *servent* envia ao receber um *ping; query* é o mecanismo primário utilizado para busca de informações na rede distribuída; *queryHit* é a resposta que um *servent* envia ao receber um *query* e *push* é um mecanismo que permite que um ponto na rede que esteja protegido por um sistema de *firewall* possa enviar e receber arquivos de outros pares.

Há diversas implementações do protocolo Gnutella que estendem os descritores originais, implementando novos recursos para ampliar as funcionalidades básicas do protocolo. No entanto, não faz parte do escopo deste trabalho o estudo dessas implementações.

2.1.6.1 Distribuição/localização de recursos

A distribuição de recursos numa rede Gnutella é feita apenas pelos pares que estão presentes na rede e detém os recursos que desejam compartilhar, ou seja, não há na especificação oficial do protocolo nenhum mecanismo que prevê o compartilhamento de recursos ou ponteiros para recursos. A localização dos recursos e dos pares na rede é feito através dos descritores *query e queryHit.*

Quando um par deseja a busca de algum recurso ele envia uma requisição *query* na rede, que possue entre seus campos no descritor da mensagem um tempo de vida na rede, o que faz com que a mensagem desapareça caso o par que a enviou não receba uma resposta

queryHit. Esse tempo de vida é relativo ao número de pares na rede que a mensagem pode atravessar buscando os recursos.

2.1.6.2 Inserção de um novo nó na rede

Não é previsto na especificação do protocolo a forma como um par entra na rede Gnutella, mas, basicamente ele deve obter o endereço IP de um par que já se encontra na rede e estabelecer uma conexão TCP/IP com esse *servent*, estando a partir daí conectado na rede Gnutella.

2.1.6.3 Exclusão de nó da rede

Depois de um tempo da saída de um nó da rede, como ele não responderá a nenhuma mensagem de *ping* com um *pong*, ele será excluído das tabelas de roteamento dos outros pares presentes na rede.

2.1.6.4 Roteamento de mensagens

É feito basicamente respeitando o caminho original que uma mensagem percorreu ao chegar num determinado *servent*, ou seja, uma resposta *pong* a um *ping*, irá percorrer o mesmo caminho do *ping*, sendo que se for enviada para outros pares na rede eles não irão propagá-la. O mesmo ocorre com os descritores *query* e *queryHit*.

2.1.6.5 Tolerância a falhas/redundância

Não é previsto na especificação básica do protocolo algum mecanismo de tolerância a falhas e redundância.

2.2 Comparação de protocolos

Como efeito de comparação, apresentamos abaixo uma tabela que compara o conjunto de protocolos estudados.

	Chord	*CAN*	*Pastry*	*Tapestry*	*JXTA*	*Gnutella*
Inserção de um nó	Inserção força a reorganização dos ponteiros locais para os nós predecessores no círculo de identificação.	Alocação de uma coordenada no espaço cartesiano para o par. Se a zona estava ocupada, é dividida ao meio.	Um novo par anuncia sua chegada a um par que seja numericamente próximo, informando de sua existência aos pares que estão na rota.	Um novo par toma o mapa de roteamento de um *gateway* conhecido.	Um novo par contacta um Super Par de Conexão como forma de atingir o restante da rede.	Um novo par contacta um *servent* presente na rede e realiza uma conexão TCP/IP com ele.
Exclusão de um nó	Exclusão força a reorganização dos ponteiros locais para os nós predecessores no círculo de identificação.	Realocação da zona ocupada para um par vizinho, que passa a ocupá-la.	O vizinho no espaço contacta o nó com maior índice ao lado do nó excluído e herda sua tabela de roteamento.	O par passa a não fazer mais parte das mensagens de atualização das tabelas de roteamento. Seu conteúdo fica temporariament e armazenado em *cache*.	Esgotando o tempo limite da existência de mensagens de publicação de recursos, os dados que foram publicados por um par somem da rede.	O nó sai das tabelas de roteamento após decorrido um intervalo de tempo em que ele pára de responder a requisições de *ping*.
Distribuição e Localização de Recursos	Chaves associadas a recursos e pares e espalhadas num círculo de identificação.	Mapeamento cartesiano para um ponto P no espaço através da chave.	Chaves associadas a recursos e pares e espalhadas num círculo de identificação.	Um servidor publica objetos roteando uma mensagem até o nós raiz. No caminho, monta um mapa em cada par no formato (servidor, objeto)	O sistema de distribuição e localização é baseado na estrutura dos Super Pares de Rendezvous.	É realizada através dos descritores query e queryHit, sendo que os pacotes possem um tempo de vida na rede.
Roteamento de Mensagens	Baseado em tabela de roteamento armazenado em cada par da rede. O ID final guia o roteamento.	Baseado no cálculo do caminho direto entre coordenadas no espaço cartesiano. Se um ponto cai no meio, outros caminhos são encontrados.	Baseado na comparação entre os prefixos dos IDs entre origem e destino.	Utiliza mapa de roteamento local que incrementa as mensagens de roteamento até o ID de destino.	Baseado na descoberta de uma rota de pares a serem percorridos, com base nas informações dos Super Pares.	É feito de *servent* em *servent* e procura respeitar o caminho original de uma determinada requisição.
Tolerância a falhas e redundância	Cada par da rede mantém localmente uma lista de sucessores de seu predecessor.	Se um par sai da rede, um vizinho assume a zona temporariamen te.	Replica os recursos aos K mais próximos pares de um par que armazena os recursos originalmente.	Copia em *cache* o conteúdo disponibilizado por um par. Utiliza tempo limite TCP e mensagens periódicas UDP.	Replicação do conteúdo dos Super Pares de Rendezvous nos pares numericame nte próximos a ele.	Não há mecanismo previsto nas especificações básicas do protocolo.

Tabela 2.1 – Protocolos P2P.

Com base na tabela acima, podemos tirar algumas informações que auxiliaram na escolha do JXTA como protocolo base para o desenvolvimento da implementação realizada neste trabalho. Inicialmente, pode-se observar que

o JXTA é o único protocolo que trabalha com uma visão contínua de que a rede P2P está em constante mutação, o que pode ser visto na forma como ele trabalha com a questão do roteamento de mensagens e da reestruturação da rede quando um par sai da mesma. Os outros protocolos necessitam de uma reorganização da topologia e atualização constante de suas tabelas de roteamento para considerarem a saída de um par em sua topologia; o JXTA simplesmente adota o recurso do tempo de vida de uma mensagem de publicação de recursos, bem como o Gnutella, excluindo dessa forma todo par que não atualizar sua publicação de mensagens depois de um intervalo dado de tempo. Outra questão é a estrutura oferecida pela abstração dos Super Pares, que permitem reduzir um fluxo de mensagens e roteamento na busca da localização de recursos e publicação de mensagens e sendo ao mesmo tempo uma estrutura descentralizada, permitindo que qualquer par na rede possa assumir esse papel; este fato não consta na especificação dos outros protocolos em questão.

2.3 Sistemas de Reputação

Oram (2001, pág. 291) apresenta uma relação de três problemas característicos de redes P2P no que se referencia com a questão da responsabilidade no sistema:

> – "A tecnologia dificulta a identificação singular e permanente dos pares e de seus operadores. As conexões e os mapas de rede podem ser transitórias. Os pares podem ser capazes de se associar e de deixar o sistema. Os participantes do sistema podem desejar ocultar informações pessoais de identificação;
> – Mesmo se os usuários possuírem um meio de identificação do par com quem eles estão lidando, eles não têm idéia de quem seja o par e nenhuma boa maneira de acessar seu histórico ou prever seu desempenho;
> – Os indivíduos que dirigem serviços par-a-par são raramente unidos por contratos e o custo e a demora de uma coerção legal geralmente seriam mais pesados que seu possível benefício."

Estudaremos aqui o funcionamento de sistemas de reputação em cinco implementações: o mecanismo promovido pelo site eBay, o mecanismo de moderação coletiva do Slashdot, protocolo Xrep, o algoritmo de EigenTrust e a métrica Advogato.

2.3.1 eBay

O eBay (www.ebay.com) é um serviço de anúncio, compra e venda de produtos pela Internet que foi lançado em 1995. O sistema funciona apenas como um intermediário de uma negociação, ou seja, um usuário que deseja vender algum produto se cadastra no sistema e o anuncia; um outro usuário que deseja comprar algum produto realiza buscas nas ofertas cadastradas e dá um lance a respeito do valor do produto.

O problema clássico aqui é como confiar num usuário vendedor, dado que o pagamento pode ser feito adiantado pelo correio ou no momento de buscar o produto. Foi por esse motivo que o eBay desenvolveu um sistema de reputação e moderação coletiva de forma que a comunidade possa destacar os bons e maus usuários do sistema conforme as trocas comerciais estiverem ocorrendo.

O mecanismo de reputação do eBay é baseado num sistema de realimentação, onde esta ocorre por parte dos usuários que finalizam uma dada transação. Após uma transação ter sido finalizada, tanto o comprador quando o vendedor podem dar notas um ao outro, além de acrescentarem algum comentário textual no histórico de transações um do outro. Essas notas consistem basicamente de três critérios: +1 (indica boa relação), 0 (neutro), −1 (indica má relação).

O mecanismo de avaliação de um usuário fornecedido pelo eBay consiste na forma de apresentar os resultados obtidos das votações que um usuário recebeu como conseqüência das transações das quais fez parte. Essas informações são apresentadas contendo as seguintes considerações: escala global, que é a soma dos votos positivos menos os negativos que o usuário recebeu;

percentual de votos positivos; o tempo em que um dado usuário já se encontra cadastrado no sistema eBay; um sumário das últimas transações realizadas pelo usuário, indicando os votos positivos, negativos e neutros que o mesmo recebeu na última semana, no último mês e nos últimos seis meses; o registro de todos os *feedbacks* que o usuário já recebeu em transações no sistema.

2.3.2 Slashdot

O Slashdot (www.slashdot.org) é um site de publicação coletiva de assuntos relacionados a tecnologia, mais conhecido como um ponto de encontro *geek* (fanáticos por computadores) na Internet. Há milhares de usuários cadastrados no sistema, que tanto podem estar lá apenas para ler as informações que são publicadas pelos outros usuários como para também escreverem seus textos e publicarem para toda a comunidade. O ponto crítico aqui é novamente a questão da moderação. Não convém para um sistema com grande número de acessos diários publicar material impróprio ou de baixa qualidade. Para resolver esse problema, o Slashdot, mais propriamente seu criador, Rod Malda, criou um sistema de moderação e reputação coletiva.

Johnson (2003, pág. 115) apresenta uma boa descrição de como o sistema funciona: "A coisa funciona da seguinte maneira: se você passou mais do que algumas sessões como usuário registrado no Slashdot, o sistema pode avisar que você recebeu o status de moderador (não muito diferente de um jurado convocado por correio). Assim como na analogia legal, os moderadores somente servem por um determinado período de tempo e, durante esse período, têm o poder de avaliar as contribuições feitas por outros usuários, em uma escala de -1 a 5. Esse poder diminui com o uso: a cada moderador confere-se um número finito de pontos que ele pode distribuir em sua avaliação. Quando todas as notas são distribuídas, seu título de moderador acaba."

São essas avaliações que formam a reputação de um usuário no sistema. Se ele é bem avaliado freqüentemente, o usuário começa a ter privilégios especiais, ou seja, passa a ser chamado mais vezes para ser moderador e seus artigos começam a ter destaque de apresentação no Slashdot.

2.3.3 Protocolo XRep

O protocolo XRep (Damiani *et al.*, 2002) é uma proposta de criação de um mecanismo auto-regulatório que possa implementar um sistema robusto de reputação numa rede P2P. O foco da implementação é para sistemas que realizam troca de arquivos.

A proposta de XRep é ser um mecanismo de avaliação não somente dos pares que compõem uma rede, mas também dos recursos por eles disponibilizados. Ele é baseado no protocolo P2P Gnutella e sua proposta é estender a especificação básica do protocolo fornecendo a estrutura para a construção e socialização da reputação baseada nas máquinas na rede e os recursos por eles disponibilizados.

Cada par na rede armazena duas tabelas. A primeira, chamada de repositório de recursos, lista cada arquivo da rede que o par já tomou conhecimento (recebeu ou enviou) e um valor binário associado dizendo se o arquivo é bom (+1) ou mau (−1). A segunda tabela, chamada de repositório de *servents*, lista os *servents* com o qual o par já trocou arquivos e o número de trocas bem sucedidas e mal sucedidas associadas.

O protocolo entra em funcionamento quando um par na rede Gnutella deseja buscar algum arquivo. Ele faz uma requisição de recurso e recebe uma mensagem de resposta dos pares que detém esse arquivo. Após receber essas mensagens de retorno, o par envia uma consulta coletiva aos pares que conhece questionando da reputação do arquivo e dos *servents* que o oferecem na rede. Os pares respondem a requisição do par que a gerou informando o resultado armazenado em suas tabelas. Após o recebimento das

reputações, o par vai computá-las e escolher o *servent* com a maior reputação e que detém o arquivo com a maior reputação associada. É a partir daí somente que a requisição de transferência do arquivo é realizada.

2.3.4 Algoritmo de EigenTrust

O algoritmo de EigenTrust (Kamvar *et al.*, 2003) tem por objetivo reduzir o número de *downloads* de arquivos inautênticos em redes P2P, associando a cada par na rede um valor de confiança único, baseado em seu histórico de *uploads*. Esse valor de confiança reflete a experiência de troca de arquivos que os outros pares da rede tiveram com o par no qual está sendo computado o valor de confiança e esse valor é ponderado pela reputação dos pares associados a ele, ou seja, os que o avaliaram.

Para iniciar o sistema, o algoritmo leva em consideração o que é chamado de pares de confiança, que são basicamente os criadores da rede P2P, pois as primeiras avaliações e ponderações são feitas baseadas nesses pares. As informações a respeito da reputação de um par são espalhadas na rede através de tabelas *hash* baseadas em protocolos como o CAN e Chord.

2.3.5 Métrica Advogato

O Advogato (www.advogato.org) é um portal na Internet que tem por objetivo servir como uma fonte de recursos para a comunidade de desenvolvedores de software livre bem como uma fonte de pesquisa e testes para métricas de reputação e confiança em trabalho em grupo utilizando a rede.

A métrica utilizada pelo Advogato é baseada em três níveis de certificação: aprendiz, intermediário e mestre. Ela analisa um conjunto de certificados de um par, resultando num conjunto de contas aceitas no *site*. Cada certificado é representado num gráfico, onde cada conta é um nó e cada certificado é uma ligação direta entre nós. A computação do nível de cada par é

feita rodando três vezes a métrica com base no nível do certificado como valor limite, ou seja, a certificação de um aprendiz é feita usando-se os três níveis e a certificação de um mestre é feita apenas usando-se certificados de mestres. Essa computação é feita, portanto, voltando-se sempre à raiz da rede, onde as contas têm mais crédito.

O fator central desse tipo de métrica é sua operação em rede, pois se há muitas interconexões o fluxo atinge praticamente todos os nós, no entanto, se há poucos nós poucos serão aceitos nessa métrica pois haverá um gargalo dos "bons" pontos da rede até as contas mais novas.

No gráfico utilizado para mapear a rede de colaboração há um parâmetro determinante: o nível de certificação do par. Cada conta corresponde a um nó no gráfico. O próximo passo é designar a capacidade de cada nó no gráfico, o que é feito levando-se em consideração a distância dos nós e a raiz do gráfico, como vemos na figura 2.1. Analisando a figura podemos dizer que, por exemplo, um determinado nó *s* irá certificar um nó *t* num determinado nível *l*, e o que nos permite verificar isso é a seta que interliga os nós.

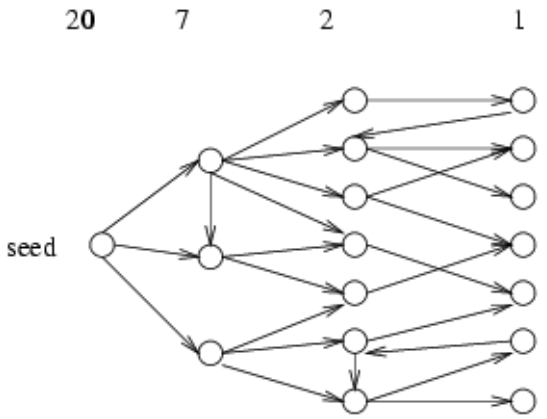


Figura 2.2 – Mapa de nós e capacidade no Advogato (fonte: http://www.advogato.org/trust-metric.html).

A objetivo concreto dessa métrica é, portanto, evitar a disseminação de nós que tenham por interesse contubar a rede e

os trabalhos de colaboração que estão sendo desenvolvidos. Ou seja, como a construção do caminho de confiança na rede é algo que passa sempre pelos nós onde a credibilidade do trabalho desenvolvido é alta e o nível de interação entre esses nós e o restante da rede que é possível escalar linearmente, temos que a rede computada dessa forma torna-se um meio de proteger os pares e a colaboração do sistema como um todo.

2.4 Aplicações e Trabalhos Relacionados

Dentro do grupo de trabalhos relacionados pesquisados, iremos apresentar aqui aqueles que serviram como base de avaliação crítica para a construção de nossa proposta, como ponto de referência daquilo que já foi desenvolvido em P2P e tecnologias associadas.

2.4.1 Iknow

O iKnow (Eikemeier *et al.*, 2003) foi projetado para ser uma ferramenta leve como suporte à colaboração baseada em redes P2P e foi implementado por uma equipe da Universidade de Bremen, usando a especificação JXTA e o projeto JAL (http://ezel.jxta.org) como ponto de partida. Seu foco de trabalho é na troca de mensagens entre os diversos membros participantes da rede, permitindo dessa forma um intenso fluxo de comunicação entre os usuários. Para tanto, o sistema oferece um mecanismo de busca de parceiros na rede com base num determinado perfil, privacidade e troca de informação de forma anônima.

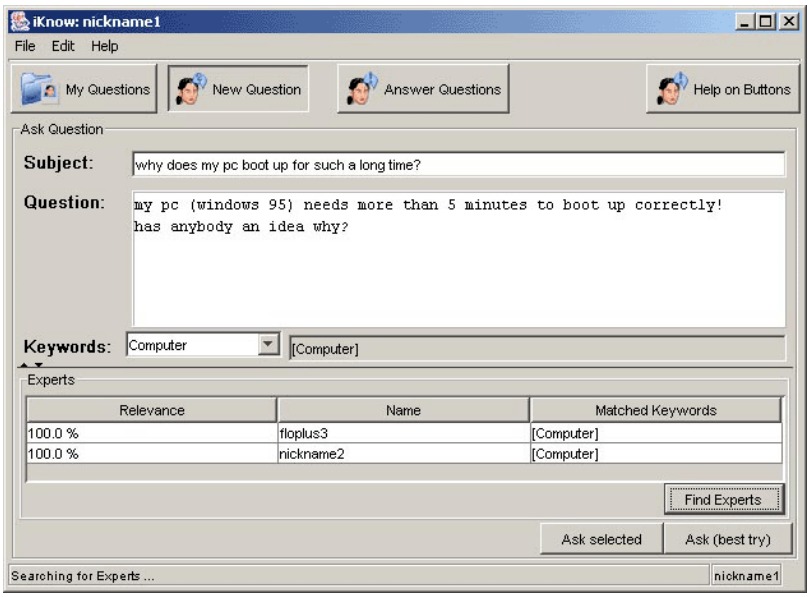


Figura 2.3 – Interface com o usuário do iKnow. (Fonte: http://iknow.informatik.uni-bremen.de/)

A ferramenta é desenvolvida de forma a motivar o usuário a construir um perfil de seu próprio conhecimento baseado em palavras chaves, que irão relacioná-lo como sendo um especialista em determinada área. Durante a operação da ferramenta, um usuário pode acrescentar ou retirar palavras dessa lista, controlando, dessa forma, e filtrando os assuntos pelos quais deseja ser referenciado na rede. Um outro usuário, no momento de formular uma questão ou no momento de escolher um usuário para trocar informações, guia-se por essas palavras chave, o que o permite obter uma possibilidade maior de resposta em uma determinada pesquisa. Quanto mais palavras chave de uma determinada consulta forem correspondentes a um perfil de um usuário, mais apto esse usuário estará a responder essa consulta; sendo assim, ele terá uma posição maior num índice relativo a uma determinada consulta.

Dessa forma, fica visível que para o bom uso dessa ferramenta, é fundamental que os usuários tenham uma fonte ampla e comum de palavras chave para o domínio de conhecimento no qual a ferramenta está inserida. Um pequeno conjunto de palavras acompanha a ferramenta; no entanto, esse é um ponto crítico de uso do sistema e que pode levar a um problema de comunicação devido à falta de padronização das palavras chave e o não

compartilhamento comum dos domínios de interesse.

O perfil de cada usuário fica residente em sua máquina local e é armazenado em disco utilizando um conjunto de dados em XML (*eXtensible Markup Language*).

2.4.2 Jabber

O Jabber (http://www.jabber.org/about/techover.php) consiste basicamente de um projeto que se propõe a ser uma referência de protocolos livres para o envio de fluxos de elementos XML entre dois pontos em uma rede e a tecnologia relacionada à construção de aplicativos utilizando esses protocolos.

A XML é parte fundamental do funcionamento do Jabber. Quando um cliente se conecta com um servidor ele está na verdade abrindo um canal de comunicação de mão dupla que consiste essencialmente de dois *streams* XML, o que permite a troca de qualquer tipo de mensagens que possa ser estruturada em *tags* XML, como textos e arquivos binários.

A arquitetura de um sistema operando com o Jabber é semelhante ao sistema de comunicação por *email*. Podemos dizer que a principal diferença consiste no fato de que o Jabber procura enviar a mensagem imediatamente se o destinatário da mesma se encontrar conectado no momento do envio por parte do emissor, fornecendo dessa forma um mecanismo que permite considerar essa comunicação como de mensagens instantâneas. Portanto, uma mensagem utilizando o sistema Jabber é inicialmente enviada a um servidor Jabber, que pode reenviar a mensagem para o destinatário se o próprio se encontrar conectado neste mesmo servidor ou abrir uma conexão com outro servidor em busca do usuário destinatário, entregando a assim a mensagem ao seu destino final. Se o destinatário da mensagem não se encontrar conectado, a informação é armazenada e posteriormente reenviada.

A comunicação somente é possível nesse sistema devido ao fato dele ser constituído de uma rede de servidores distribuídos, basicamente como o sistema de *email*. O roteamento de mensagens entre esses servidores é feita baseada no sistema de DNS, o que somente é possível devido ao fato de Jabber utilizar um sistema de endereçamento entre pares baseado nos esquemas URI, ou seja, no formato *user@host*.

Esse esquema de endereçamento utilizado é chamado de JID (*Jabber Identifier*) e é composto por uma série de elementos ordenados e possui o seguinte formato: nó@domínio/recurso. A identificação de domínio é a identificação primária e representa o servidor Jabber ao qual o par está conectado. O identificador do nó representa a identificação do usuário propriamente dito. O recurso é um identificador opcional que permite que um par possa manter muitas conexões simultâneas com um mesmo servidor Jabber.

Podemos dizer, portanto, que o Jabber utiliza a arquitetura cliente / servidor para sua operação normal, com a vantagem de ser aberta a qualquer par a possibilidade de ser um servidor Jabber, podendo eventualmente, conectar dois pares diretamente num caso de troca de arquivos, mas mesmo essa conexão direta é mediada pelos servidores em primeira instâncias, pois são eles que informam quais usuários estão conectados num dado momento e qual seus endereços para contato direto.

Os servidores Jabber são construídos de forma modular e desempenham essencialmente três funções: tratar as conexões dos clientes e permitir a comunicação direta com os clientes Jabber, evidenciando sua face P2P, comunicar com outros servidores Jabber, coordenar os diversos componentes modulares instalados no servidor. São esses componentes modulares que são os responsáveis pela autenticação de usuários, listas de contato, armazenamento de mensagens enviadas quando um usuário está desconectado e a interoperabilidade com outros sistemas de mensagens instântaneas, como o ICQ.

Por ser um projeto baseado em protocolos livres, há diversas implementações (www.jabbercentral.com) de clientes Jabber disponíveis para os mais variados sistemas operacionais. Como forma de ilustrar uma das implementações, na figura 2.4 podemos ver o WinJab, cliente Jabber para o sistema operacional Windows.

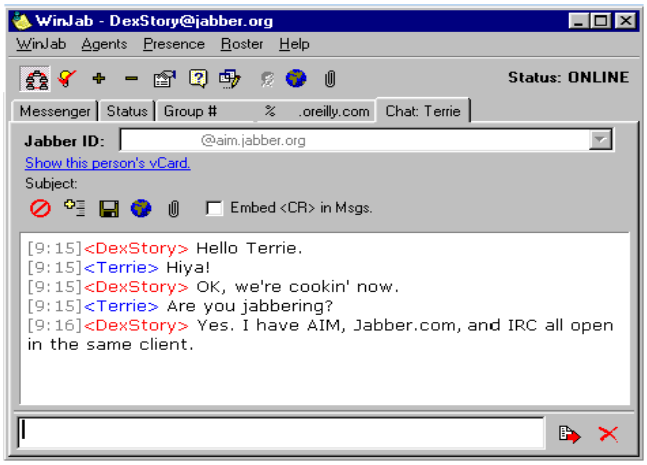


Figura 2.4. WinJab – Tela de *chat* com outro par. (Fonte: http://www.openp2p.com/pub/a/p2p/2000/10/06/jabber_works.html)

2.4.3 ICQ

O ICQ (*I Seek You*) é um dos aplicativo de mensagem instantânea mais difundidos entre os usuários da Internet da atualidade. O sistema possui um protocolo proprietário para a comunicação entre usuários e sua arquitetura de funcionamento possui uma forma híbrida entre o P2P e cliente/servidor.

Logo no momento de instalação do aplicativo é demandado ao usuário para criar seu nome de conta e senha, recebendo em seguida o usuário um número de registro que será sua identificação na rede ICQ. Esse número de identificação é registrado nos servidores do sistema ICQ, que também são proprietários e não distribuídos livremente, e permite que o usuário possa ser encontrado mesmo quando esteja desconectado devido ao fato dos servidores armazenarem seu perfil, número de identificação e estado de uso.

Qualquer comunicação utilizando a rede ICQ é mediada, desta forma, pelos servidores. Um usuário constroe no sistema uma lista de contatos (ver figura 2.5), que representa as pessoas com as quais ele tem conhecimento do número de registro e logo pode manter contato. Quando um usuário trocar dados com outro, uma requisição é enviada ao sistema de servidores requisitando o número IP desse outro usuário. A partir disto, uma conexão TCP é estabelecida diretamente entre os dois usuários e o servidor deixa de mediar a conversação. É nesse ponto que o sistema torna-se P2P. Mais informações sobre a rede ICQ podem ser encontradas em http://www.student.nada.kth.se/~d95-mih/icq.

Após estabelecida uma conexão um usuário tem a possibilidade de trocar mensagens com um outro par na rede, enviar e receber arquivos, enviar e receber endereços de *sites* para consulta na Internet, abrir uma sala de chat com múltiplos usuários, requisitar contatos de outros usuários e ver as informações de perfil registrada por um determinado usuário no sistema de servidores.

Figura 2.5. Lista de contatos e funcionalidades ICQ. (Fonte: www.icq.com)

O fato do ICQ usar uma estrutura de servidores que gerenciam as conexões entre usuários permite que um determinado cliente do sistema possa receber mensagens que lhe foram enviadas mesmo quando ele se encontrava no estado desconectado, o que somente ocorre pois o servidor armazena essas mensagens e assim que o estado de usuário conectado é detectado, as mensagens são enviadas.

Outra característica oferecida pelo sistema de servidores do ICQ é a possibilidade de ter como recurso uma interface de busca de usuários no sistema, exibida na figura 2.6. Essa interface permite que uma busca por um usuário possa ser feita por campos-chave, como nome do usuário, empresa

onde trabalha, ocupação, *email*, línguas que fala, entre outros. A busca é realizada nos bancos de dados dos servidores e se é bem sucedida, o usuário que fez a requisição recebe o número de registro da pessoa que desejava encontrar. A partir desse momento, ele pode enviar uma requisição para incluir esse usuário em sua lista de contatos. Essa requisição pode ser automaticamente aprovada ou passar por uma moderação do usuário requisitado, dependendo das configurações de seu perfil.

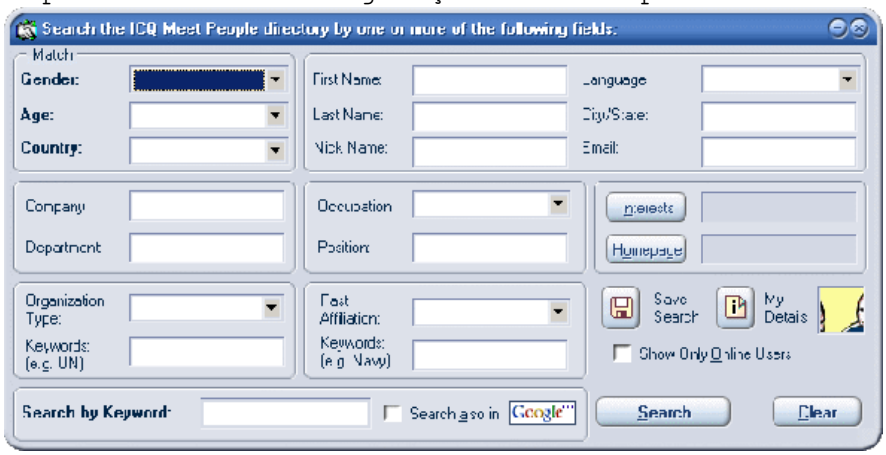


Figura 2.6. Interface de consulta de usuários ICQ. (Fonte: http://www.icq.com)

2.4.4 Edutella

Edutella (Nejdl, 2002) é um projeto que tem por objetivo especificar e implementar uma estrutura de metadados baseada em RDF(*Resource Description Framework*) para o projeto JXTA. A idéia final é obter uma forma padronizada, que pode ser proporcionada pela adoção generalizada dos metadados, de disponibilização, busca e anotação sobre os recursos que podem ser oferecidos na rede. O projeto visa inicialmente prover quatro serviços de referência: serviço de busca, para fornecer a estrutura necessária para a recuperação de dados em formato RDF na rede P2P; serviço de replicação, para promover a persistência e disponibilidade dos dados, bem como manter o balanceamento de carga nos diversos nós da rede que detém um determinado recurso, mantendo ao mesmo tempo a integridade e consistência dos dados; serviço de mapeamento, que permite a tradução entre diferentes vocabulários de metadados, permitindo a interoperabilidade entre os diversos pares na rede; serviço de anotação, que permite criar anotações

em qualquer recurso que possa estar disponibilizado nos pares da rede.

De forma geral, o foco do projeto Edutella é criar uma infra-estrutura de interoperabilidade entre os mais diversos recursos e sistemas que possam usufruir das características de uma rede P2P desenvolvida através da especificação JXTA. Não tem como foco, necessariamente, a colaboração efetiva entre os mais diversos pares e as possibilidades de interação que podem ser geradas do encontro de pares. Isto quer dizer que o sistema não é centrado na troca de fluxos de conversação entre pares, que é ponto principal de colaboração onde nos baseamos neste presente trabalho.

Até o momento presente em que esses estudos foram realizados, não tivemos notícia de um cliente que houvesse sido implementado com as especificações propostas pelo projeto Edutella.

2.4.5 LionShare

O LionShare (Halm, Valentine, 2002) consiste basicamente de um sistema P2P com foco na área acadêmica que tem por objetivo colaborar com a distribuição de publicações acadêmicas entre as universidades e outras instituições. Ele é um sistema de código livre, sendo construído com base no projeto Limewire (Rohrs, 2001), escrito em Java, sendo um dos mais famosas propostas disponíveis baseado na especificação de protocolos Gnutella. No entanto, algumas alterações foram promovidas de forma a permitir que os pares do sistema tenham de inicialmente que se autenticar, de forma a evitar que um usuário possa participar da rede LionShare de forma anônima.

Por ser feito em Java, o LionShare é multiplataforma. Uma característica relevante do projeto é o fato dele permitir que o usuário possa escolher se deseja compartilhar seus arquivos via arquitetura cliente/servidor ou P2P. Dessa forma, publicar arquivos num servidor tem praticamente o mesmo efeito que disponibilizar essas arquivos em disco local, na máquina de trabalho do usuário.

O LionShare se propõe a usar intensamente metadados, de forma a possibilitar uma busca específica pelo tipo de mídia a ser pesquisado, como vídeo ou áudio. Outra proposta do projeto é permitir a anotação em arquivos

de mídia utilizando XML, possibilitando dessa forma uma interface para os usuários anotarem comentários sendo possível resgatar através de consultas as informações registradas.

O foco do LionShare é basicamente na troca de arquivos e disponibilização de recursos através das redes P2P. O projeto deixa em aberto questões como o perfil de usuários na rede, formas de encontrar pessoas através de consultas específicas e a reputação dos usuários do sistema.

Segundo o site oficial do projeto (http://lionshare.its.psu.edu), ainda não há uma implementação oficial do sistema.

2.5 Comparação de sistemas

Iremos, seguindo os quatro critérios abaixo especificados, analisar comparativamente os sistemas acima descritos.

Critério 1 – Arquitetura de interação: consiste em analisar a arquitetura de rede proposta como canal de interação entre os usuários participantes do sistema.

Critério 2 – Interface e usabilidade: consiste em analisar a interface gráfica oferecida como mediação entre as funcionalidades oferecidas pela arquitetura de rede e o usuário final.

Critério 3 – Portabilidade: consiste em analisar o nível em que a solução pode ser utilizada em diversas plataformas operacionais.

Critério 4 – Protocolos: consiste em analisar qual o protocolo adotado pela solução e seu impacto no resultado final.

	Iknow	*Jabber*	*ICQ*	*Edutella*	*LionShare*
Arquitetura de Interação	Utiliza os recursos de interação do JXTA, permitindo portanto descentralização e comunicação P2P através do uso dos Super Pares.	Utiliza um sistema de servidores distribuídos como mecanismo de interação entre usuários.	Rede de servidores que centralizam o registro de usuários e cadastro de perfis. Estabelecimento de conexão pode ser feito diretamente entre usuários, após mediação do servidor	Utiliza os recursos de interação do JXTA, permitindo portanto descentralização e comunicação P2P através do uso dos Super Pares.	P2P descentralizado.

Interface e Usabilidade	Cada funcionalidade tem sua janela de interação própria, focando o sistema nas funcionalidades: troca de arquivos, chat, questões.	Há vários clientes disponíveis para o sistema e focam interface nas funcionalidades do sistema: essencialmente, troca de mensagens.	A interface tem o foco na lista de pares conectados ativos na rede: troca de mensagens, arquivos, chat coletivo, troca de endereços *web*.	Implementação não disponível.	Focado nas funcionalidades do sistema: troca de arquivos e anotação de arquivos de mídia usando XML.
Portabilidade	Desenvolvido usando linguagem JAVA.	A especificação se utiliza do XML, podendo ser desenvolvido clientes em qualquer linguagem que suporte interação XML.	Clientes proprietários e oficialmente somente para Windows e Macinstosh.	Implementação não disponível.	Desenvolvido usando linguagem JAVA.
Protocolos	JXTA	Jabber, especificação aberta a comunidade.	ICQ, proprietário.	JXTA	Gnutella, modificado para autenticação.

Tabela 2.2 – Comparação entre aplicações.

Analisando a tabela acima, podemos notar algumas características interessantes entre os sistemas considerados. Primeiramente, apenas o ICQ tem como foco principal de interação com o usuário do sistema uma lista de pares e desta lista derivando as funcionalidades que o sistema pode oferecer, sendo que os outros ambientes possuem o foco de sua interface nas funcionalidades oferecidas. Tanto o Jabber como o ICQ utilizam um sistema híbrido, baseado em uma estrutura prévia de mediação por servidores e após uma conexão direta entre usuários. No entanto, os servidores do sistema ICQ são proprietários e do Jabber são descentralizados permitindo uma maior disseminação e socialização dos recursos de servidores e mediação. O LionShare e o iKnow são desenvolvidos em JAVA, permitindo uma maior interoperabilidade entre os diversos sistemas operacionais que suportam JAVA e o cliente implementado para esses sistemas.

2.6 Resumo

O estudo e a análise dos diversos protocolos e sistemas relacionados levou a compreensão das estruturas tecnológicas de mediação e colaboração em redes P2P. De forma comparativa pudemos analisar diversos protocolos que permitem a criação da infraestrutura lógica básica para a interação entre os diversos pares presentes num sistema e pudemos sinalizar para a especificação JXTA como a mais adequada para a implementação de um sistema focado na colaboração entre um pequeno grupo de usuários com interesses em

comum e com um foco específico de colaboração num ambiente em princípio multiplataforma.

Também de forma comparativa pudemos analisar cinco projetos que se utilizam de recursos P2P como possibilidade de conexão entre pares do sistema. No entanto, esse estudo nos levou a considerar uma implementação que pudesse ter como foco de interação com o usuário a lista de pessoas conectadas no sistema de forma a evidenciar a presença do "outro" na rede e, também, utilizando um cliente e um sistema de protocolos que fosse essencialmente independente de um sistema de servidores proprietários e interoperável entre diversas plataformas. É essa linha de projeto que foi adotada na especificação do sistema proposto e implementado, descrito no capítulo 3.

Capítulo 3 – O Sistema de Colaboração em P2P: SemiCode

Tendo como ponto de partida o estudo dos protocolos e sistemas descrito no capítulo 2, podemos agora definir e especificar a proposta de desenvolvimento de um sistema computacional, daqui por diante denominado SemiCode, que possa servir como suporte à colaboração num ambiente de rede P2P.

Realizar esse exercício, o de definir e especificar o sistema SemiCode, é uma tarefa que passa por estabelecer quais são os pontos chaves em termos conceituais que devem nortear o processo de especificação e desenvolvimento, e quais são os recortes necessários para a implementação de um grupo de funcionalidades coerente e catalizador da colaboração entre pares. Como o primeiro recorte e como aspecto fundamental por parte da implementação do sistema podemos considerar a infra-estrutura de rede, ou seja, o grupo de protocolos que vai permitir que a aplicação possa construir sua camada de interação sobre uma topologia de rede efetivamente P2P. A escolha pelo JXTA e por sua implementação em Java, que passou pela questão de utilizar um sistema de rede P2P que fosse interoperável entre diversas plataformas e permitisse uma estrutura de rede que não utilizasse os recursos de servidores específicos, ficou descrita e justificada no capítulo 2.

O segundo ponto chave para o desenvolvimento da aplicação é a forma de mediação entre a infraestrutura de rede e o usuário. Por essa interface passam questões como a usabilidade do sistema, as estruturas técnicas que vão permitir a real colaboração entre pares na rede e a capacidade de captar a interação entre os efetivos recursos de rede e as possibilidades oferecidas aos pares. A questão principal que aqui colocamos é que a proposta de desenvolvimento da forma de mediação do SemiCode tem de necessariamente atender a demanda de que o sistema está focado no encontro do outro na rede, no encontro de um outro par e, a partir desse encontro, selecionar um par de um conjunto de pares presentes na rede com o qual pretende-se trabalhar, tendo-se, desse ponto em diante, disponível um

conjunto de funcionalidades que possam concretizar a colaboração no ambiente P2P. Como forma de atender a essa demanda, projetamos um mecanismo que possa criar uma lista de companheiros ativos na rede e a partir dessa lista selecionar o par com o qual desejamos trabalhar. A partir desse momento, podemos iniciar a troca de informação com esse par, o que nos permite dizer que a interface de relacionamento com o usuário no SemiCode estará centrada no encontro de pessoas e não nas funcionalidades que o mesmo pode disponibilizar.

O terceiro ponto chave na especificação do sistema está relacionado com as funcionalidades que o SemiCode oferece como possibilidade de interação entre pares. A escolha das funcionalidades a serem apresentadas aos pares está relacionada com a questão central do encontro do outro na rede. Esse encontrar o outro é traduzido em notar a presença de um determinado par na rede, o que está representado no sistema pela lista de pares ativos num dado momento; poder trocar informações com esse par, ou seja, estabelecer uma conversação dinâmica e síncrona com ele; enviar questões relativas ao escopo dos trabalhos desenvolvidos pelos pares que possam ser respondidas de forma assíncrona, ou seja, sem a preocupação de uma resposta instantânea pelo par que a recebe; ver e trocar trabalhos produzidos pelos pares e entre os pares e, por final, descobrir quem é determinado par dentro do contexto da rede P2P e como ele é visto pelos demais pares do sistema, através da possibilidade de desenvolvimento e publicação de um sistema de reputação descentralizado e colaborativo.

São, desta forma, essas quatro funcionalidades que compõem as possibilidades de interação entre pares do SemiCode: *chat,* troca de arquivos, envio e resposta de questões e reputação. No entanto, vale destacar que essas funcionalidades são apenas uma amostra do tipo de interação que podemos ter no âmbito do sistema. Se, no decorrer do processo de desenvolvimento e manutenção do sistema, desejarmos incluir um novo módulo, como videoconferência por exemplo, basta utilizarmos a estrutura orientada a objetos do sistema para que ele faça parte dos processos de interação entre pares. Mas, nesse primeiro momento, iremos iniciar o sistema com as quatro funcionalidades acima descritas.

Portanto, juntando-se a forma de mediação com os pares da rede e as funcionalidades propostas, temos os pontos de partida conceituais que nos servirão de referência durante o desenvolvimento da proposta de trabalho e das implementações realizadas.

3.1 Descrição Conceitual do Sistema SemiCode

Estabelecidos os parâmetros iniciais de referência conceitual do SemiCode, podemos agora nos deter mais atentamente na descrição dos módulos que irão compor o sistema e que irão implementar as funcionalidades e dinâmicas básicas de interação. Podemos observar na figura 3.1 um diagrama representando esses módulos e a interação entre eles.

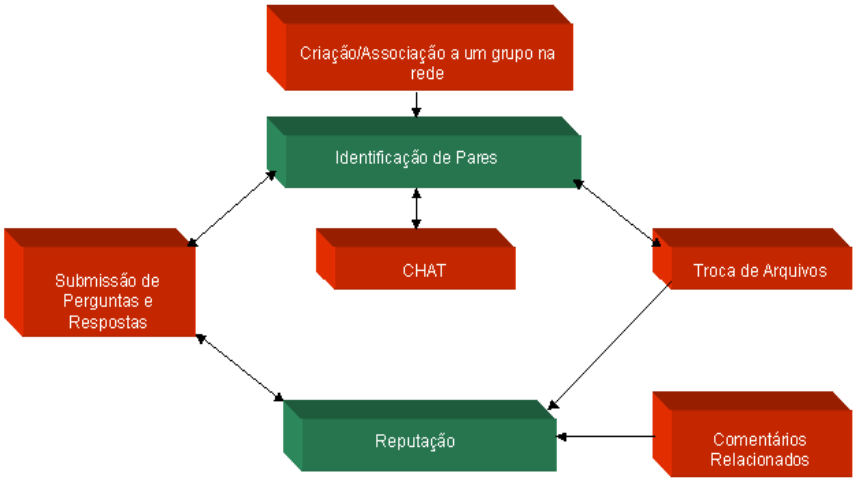


Figura 3.1 – Módulos Conceituais do SemiCode.

A figura 3.1 além de denotar os módulos que compõem o sistema SemiCode, evidencia a relação de funcionamento entre eles que carrega em sua composição a preocupação central de propiciar o encontro de um outro par na rede. Isso fica evidente no primeiro e segundo módulo da figura, criação/associação a um grupo na rede e identificação de pares,

respectivamente, onde um par inicialmente vai se associar ou criar um grupo para a colaboração entre pares e num segundo momento esse par vai receber uma listagem de pares conectados e associados ao mesmo grupo. É desse ponto em diante que se estabelece o encontro dos pares no sistema, dado que os mesmos já foram identificados e a sua presença comunicada ao usuário do sistema.

Vamos agora especificar os módulos do sistema.

O módulo de Criação/Associação a um grupo na rede tem por finalidade inicializar e inserir o usuário do sistema Semicode como um par ativo na rede P2P. Ele é reponsável por inicializar os serviços de rede, o que consiste da criação e associação a um grupo de pares. Esse grupo de pares consiste do grupo onde a aplicação terá o seu foco de uso. Como mencionado no capítulo 1, esse grupo é caracteristicamente um grupo com interesses gerais semelhantes, como alunos de um mesmo curso, pesquisadores de um mesmo projeto, professores de um mesmo departamento, enfim, um pequeno grupo de usuários que partilham uma realidade de trabalho em comum e que utilizarão o SemiCode como suporte ao desenvolvimento do trabalho colaborativo. Todos os usuários desse grupo devem inicializar o aplicativo da mesma forma, ou seja, com o mesmo nome de grupo para poderem se encontrar dentro do mesmo grupo de pares.

O Módulo Identificação de Pares é o responsável por identificar os pares ativos dentro de um mesmo grupo de pares e informar ao usuário quais estão conectados no momento de uso do sistema. Uma questão fundamental para o funcionamento adequado desse módulo é que o mesmo monitore periodicamente a rede para detectar a inserção e a saída de pares da rede. É este módulo o responsável pela construção dinâmica da lista de usuários conectados no grupo através do SemiCode. É importante notar que esse módulo é o responsável conceitualmente apenas pelo monitoramento da rede e que a forma como isso pode ser apresentado ao usuário final é bastante flexível e pode ser adaptável a diversas metáforas, desde a criação de uma lista de pares ativos, como o sistema ICQ funciona, até a representação gráfica de avatares (personificação) dos pares numa metáfora de sala de encontro, mesa

de bar, sala de aula, enfim, as mais variadas possibilidades de forma a atender demandas pedagógicas e culturais.

O Módulo Chat tem por finalidade apresentar uma interface para conversação entre dois pares na rede de forma dinâmica e síncrona. Dado que um par encontra-se ativo na rede, podemos chamá-lo para uma conversa e estabelecer um relacionamento de troca de informações com o mesmo, baseadas essencialmente em troca de informações textuais.

O Módulo Troca de Arquivos oferece uma interface para um usuário do sistema identificar quais são os arquivos que um outro par na rede disponibilizou para acesso público aos pares presentes no mesmo grupo. Estes arquivos, além de visualizados por um usuário do sistema, podem ser transferidos da máquina do par que os gerou para a máquina do usuário que observa a lista. O objetivo dessa transferência é fornecer uma grande base de arquivos armazenados de forma descentralizada e de acesso público, permitindo dessa forma que um usuário possa ler documentos que são partilhados na rede, descobrir textos de referência que um outro usuário disponibilizou, acompanhar o desenvolvimento de documentos, participar de processos de produção colaborativa de documentos e socializar qualquer informação que o usuário do sistema achar adequado e conveniente para a rede.

Temos aqui um ponto crítico inerente a própria tecnologia P2P: não há moderação sobre o tipo e o conteúdo dos arquivos que são disponibilizados publicamente para acesso de qualquer par presente na rede. A idéia do SemiCode é exatamente essa, ou seja, deixar a moderação do conteúdo disponibilizado por parte do usuário do sistema. No entanto, como veremos na seção 3.1.6, prevemos um mecanismo de avaliação coletiva de um par presente na rede, que tem por objetivo servir como mediação coletiva do trabalho produzido por esse par e dos recursos disponibilizados pelo mesmo, além de também servir como forma de mapear usuários que se destacam na colaboração e desenvolvimento coletivo da rede P2P. É através desse mecanismo que se pretende coibir coletivamente a disponibilização de arquivos impróprios, tentativas de inclusão de vírus na rede e disponibilização de conteúdo com restrições de propriedade intelectual para

difusão e publicação pública.

O Módulo de Submissão de Perguntas e Respostas é o responsável pelo gerenciamento do sistema de submissão de perguntas e respostas do SemiCode. A idéia por trás deste módulo é oferecer aos usuários do sistema uma interface onde os mesmos possam enviar questões para outros pares e responder questões recebidas de outros pares, mas de forma assíncrona, ou seja, sem a necessidade de estabelecer uma conversa através do módulo de *chat*. Este módulo simula de uma certa forma um sistema de *emails* dentro do ambiente da rede P2P e de uma sessão de uso, pois após a desconexão do usuário as questões recebidas e enviadas não são armazenadas e descartadas automaticamente como mensagens já respondidas. O conceito que pretendemos abordar aqui é de não poluir excessivamente uma sessão de uso do SemiCode com muitas requisições de informações por parte dos usuários através de tentativas de estabelecimento de conversações. Com este módulo, os usuários podem verificar seu módulo de questões no momento em que desejarem para averiguar se receberam respostas de questões enviadas ou se receberam perguntas aguardando respostas. Também cabe aqui a questão de que a moderação deste módulo cabe a cada usuário do sistema, lembrando que temos um sistema de avaliação da reputação de cada usuário, que pode ser melhor avaliado conforme sua participação for mais ativa na rede, o que pode ser traduzido por responder constantemente as questões que recebe e socializar seu conhecimento.

O Módulo de Reputação e Comentários Relacionados é o responsável pelo controle e operação do sistema de reputação utilizado pelo SemiCode. Falar em reputação em sistemas de informação é tratar de um assunto ainda bastante polêmico e em constante experimentação. O que pretendemos aqui é iniciar essa discussão da moderação coletiva e do mapeamento da forma como a comunidade de pares de uma rede P2P avalia a si própria.

Com base nas análises dos sistemas descritos no capítulo 2, tomamos duas diretrizes de projeto na especificação do sistema de reputação do SemiCode: o sistema deve ser realimentado pela comunidade de pares, ou seja, é necessário registrar o *feedback* gerado pela comunidade e o sistema de pontuação deve ser relativo e não apresentar apenas um índice absoluto que

pouco informaria a respeito da reputação de um par na rede.

Portanto, o que propomos como um sistema de reputação para o SemiCode consiste basicamente em: registro do perfil do usuário do sistema em arquivo (o preenchimento será opcional por parte do usuário), contendo informações como seu nome, preferências pessoais e aptidões profissionais; avaliação remota de um par contendo três possibilidades de avaliação – par colaborativo, normal ou ausente – que irão servir para compor uma avaliação da forma como esse par é visto pela comunidade; registro de comentários de outros pares.

Dessa forma, o sistema pode servir como fonte de maiores informações a respeito de um par na rede, registro de comentários a respeito das relações de outros pares com um par específico na rede e um sistema de avaliação que pretende captar a atuação desse par na rede P2P. É através desse mecanismo que pretendemos promover a moderação coletiva, evitando a disseminação de conteúdo inadequado na rede e estimulando a colaboração entre pares.

3.2 Modelagem do Sistema SemiCode

Dado que estabelecemos a descrição conceitual e os módulos operacionais do sistema, podemos agora discutir a forma como essa descrição pode ser modelada de forma a tornar-se um sistema computacional e criar diretrizes de sistemas que possam auxiliar e guiar o processo da implementação dos conceitos aqui descritos e especificados.

Para a realização de tal tarefa, iremos nos guiar por algumas etapas e especificações do processo ICONIX (Rosenberg e Scott, 2001), que consiste num sistema de modelagem não tão complexo quanto o RUP (*Rational Unified Process*) e não tão simples como o XP (*Extreme Programming*). A idéia é fornecer o subsídio para a construção de algumas etapas de modelagem de forma a auxiliar no processo de documentação e implementação de um sistema computacional. Iremos apresentar aqui três etapas que foram fundamentais no mapeamento da descrição conceitual do sistema para uma especificação que pudesse servir como guia e referência constante dos trabalhos de implementação que foram desenvolvidos e são descritos no capítulo 4: descrição textual do domínio do problema, casos de uso e diagrama de

classes.

3.2.1 Descrição Textual do Domínio do Problema

Esta descrição tem por objetivo descrever o comportamento do sistema e a relação com o usuário de forma geral. É deste exercício que podemos "descobrir as classes que representam as coisas e os conceitos relacionados ao problema para o qual o sistema é designado a resolver" (Rosenberg e Scott, 2001). Segue abaixo a descrição do SemiCode.

"Ao iniciar o uso do sistema, o usuário é questionado a respeito do seu nome de usuário e senha, dados que serão validados de forma a permitir a inicialização dos serviços de rede P2P. Esse nome e senha são gerados quando da instalação do pacote JXTA (www.jxta.org), o qual é necessário para o posterior uso do SemiCode no acesso aos serviços de rede. A janela de interação que recebe essas informações é gerada automaticamente pelo pacote JXTA quando da inicialização dos serviços de rede pelo SemiCode.

Após a autenticação no sistema, o usuário vai ter acesso a interface gráfica do mesmo, onde será exibida uma listagem dos usuários conectados no momento naquele domínio de rede P2P onde o SemiCode está sendo utilizado. Essa listagem é constantemente atualizada conforme a entrada ou saída de pares da rede, permitindo assim que o usuário do sistema tenha sempre uma visão atualizada dos pares que estão conectados.

Após a listagem de pares ser exibida, ao usuário do sistema será oferecida a possibilidade de selecionar um determinado par presente na listagem de pares conectados e arrastá-lo para área de trabalho. A partir desse movimento, o usuário tem a sua disposição os serviços de interação com o par selecionado: abrir um canal de chat para a troca de mensagens, ver os arquivos que o par disponibiliza publicamente para que todos os pares possam ter acesso, enviar questões ao par, ver sua reputação e o perfil associado ao par selecionado para trabalho.

Para cada usuário do sistema, tem-se a possibilidade de configuração do SemiCode na interface principal, de modo que esse usuário possa entrar com os dados pessoais relativos ao seu perfil na rede, que serão sempre enviados aos outros pares quando da requisição de dados sobre a reputação deste par, e da escolha do diretório onde irão ser armazenados os arquivos que serão disponibilizados para troca com os outros pares."

É com base na descrição acima que podemos passar para a próxima etapa da modelagem do sistema, que consiste da especificação dos casos de uso, que são descrições dos cenários de uso do sistema pelo usuário de forma detalhada.

3.2.2 Casos de Uso

A modelagem do sistema por casos de uso tem por objetivo capturar as necessidades dos usuários em um novo sistema através do detalhamento dos cenários de uso do sistema que o usuário irá encontrar em seu cotidiano (Rosemberg e Scott, 2001). É, portanto, um exercício no esforço de estabelecer, passo a passo, os caminhos que o usuário irá percorrer em seu relacionamento com o sistema objetivando a interação entre os pares na rede. Dessa forma, representaremos aqui os casos de uso através do diagrama UML e uma descrição textual pontuando os atores do sistema e as tarefas desempenhadas pelo mesmos.

Inicialmente, podemos observar na figura 3.2, temos que o usuário do sistema SemiCode irá realizar a operação de se identificar no sistema. Esse processo de identificação passa por duas etapas, onde a primeira está relacionada ao registro do usuário no sistema SemiCode, que está basicamente ligado ao perfil desse usuário na rede. Tanto que se o nome do usuário não se encontrar registrado no SemiCode instalado localmente, o mesmo é levado a preencher as informações relacionadas ao seu perfil, o que pode ser visto no caso de uso da figura 3.3. Após preencher os dados de registro no Semicode, o usuário é levado a preencher as informações relativas a

seu nome de usuário e senha da rede JXTA.

Aqui, vale fazer um aparte da discussão de casos de uso para explicitar essa questão do registro de usuário na rede JXTA.
Quando o usuário instala o sistema JXTA em sua máquina local, ele recebe nos pacotes de instalação alguns códigos de exemplo, um *shell* interativo para exercitar comandos que operem numa rede P2P com base nos protocolos JXTA e um configurador que permite ao usuário ter acesso aos recursos dessa rede. Esse configurador demanda do usuário informações a respeito de seu número IP local, porta onde o mesmo será instalado, informações de segurança, etc. Toda vez que um usuário for utilizar qualquer serviço da rede JXTA ele precisa validar seu registro em relação a instalação local. Essa etapa de configuração é feita antes do sistema SemiCode estar rodando ou mesmo instalado na máquina, portanto o sistema apenas inicializa a rede P2P e apresenta a interface que permite ao usuário se autenticar na rede P2P.

É esse fato que justifica as duas informações de registro demandadas pelo sistema SemiCode ao início de suas operações.

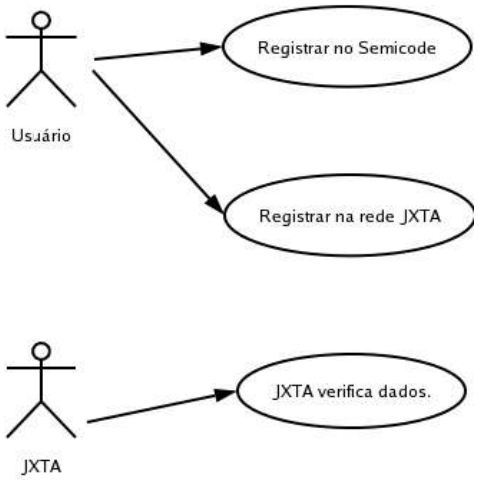


Figura 3.2 – Pacote de Casos de Uso: Usuário Logando no Sistema.

Logo após a identificação do usuário no SemiCode e na rede JXTA, o sistema verifica se o usuário já possui seu perfil de dados e

opções pessoais registradas, o que podemos ver representado na figura 3.3. Se não houver registro, o sistema apresenta de imediato a interface que permita que o usuário informe suas opções relativas ao seu perfil pessoal. Essas informações são armazenadas em disco em um arquivo e podem ser visualizadas ou alteradas pelo usuário do sistema a qualquer momento. O fato de termos essas informações armazenadas em disco permite que no decorrer de uma sessão de uso do SemiCode usuários remotos possam visualizar o perfil de um determinado par na rede, o que permite obter mais informações a respeito dos outros pares presentes na rede.

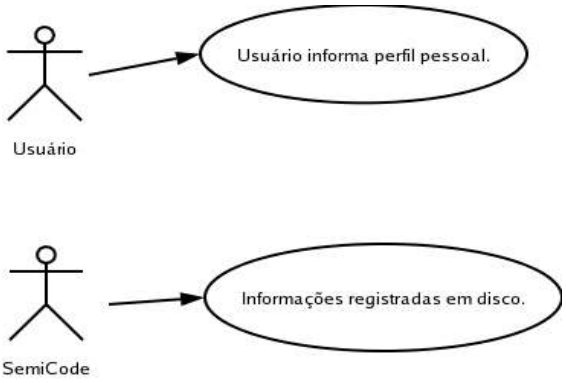


Figura 3.3 – Pacote de Casos de Uso: Usuário Preenche seu Perfil.

Após se registrar no sistema e ter validadas suas informações de usuário, o sistema SemiCode fará uma varredura na rede P2P em busca dos pares que também estão conectados, conforme figura 3.4. Essa varredura é feita de forma constante, o que permite ao sistema sempre atualizar os novos pares que entram na rede e os pares que saíram do sistema.

Esse processo de varredura em busca das informações de quem está conectado na rede apresenta ao usuário do sistema uma listagem de pares e, a partir disto, ele tem a possibilidade de selecionar um par e iniciar uma sessão de trabalho com o mesmo. Nessa sessão de trabalho o par pode trocar informações via *chat*, ver e utilizar os arquivos disponibilizados para compartilhamento pelo par remoto,

enviar e responder questões e visualizar ou avaliar a reputação e perfil de um determinado usuário. Toda a vez que o usuário do sistema quiser trabalhar com outro par ele inicialmente precisa selecioná-lo da lista de pares conectados e assim ter acesso aos serviços oferecidos pelo sistema.

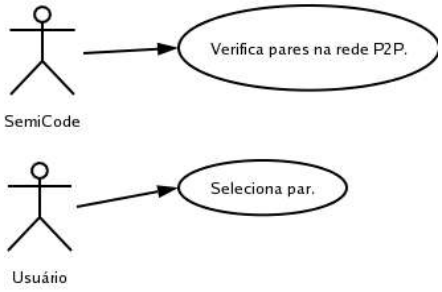


Figura 3.4 – Pacote de Casos de Uso: Usuário Seleciona um par na rede para trabalhar.

Tendo o usuário do sistema selecionado o par com o qual ele deseja trabalhar da lista de pares conectados, cabe ao mesmo selecionar o tipo de atividade, ou seja, o tipo de interação que será estabelecida com o par remoto. É importante aqui ressaltar que o sistema foi modelado de tal forma a permitir que um processo técnico de interação possa funcionar de forma modular, ou seja, podemos no processo de desenvolvimento do sistema acrescentar outras possibilidades de interação com a mínima necessidade de alteração de código e mudança nos módulos já implementados. Essa possibilidade nos permitiria acrescentar, por exemplo, um módulo de vídeoconferência.

Na figura 3.5, podemos visualizar o caso de uso para quando o usuário do sistema seleciona a opção de bate papo com o par remoto. Nesse processo, o usuário pode digitar mensagens e enviá-las para o par e o sistema SemiCode cuida do monitoramento da chegada de novas mensagens do par remoto em resposta as enviadas pelo usuário do sistema.

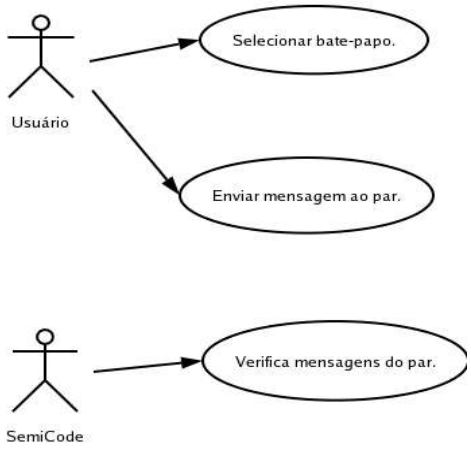


Figura 3.5 – Pacote de Casos de Uso: Usuário interage pelo bate papo com um par selecionado.

Na figura 3.6, podemos visualizar o caso de uso relacionado ao momento em que o usuário do SemiCode resolve verificar ou avaliar a reputação de um determinado par na rede. Esse processo consiste basicamente na exibição para o usuário do sistema das informações de perfil de um determinado par e como esse par tem sido avaliado por seus outros pares na rede. Nesse momento, o usuário em questão tem a possibilidade de avaliar o par conforme os parâmetros já descritos no item 3.1.6. O resultado dessa avaliação ficará armazenado na máquina local do usuário que foi avaliado, permitindo assim que outros pares na rede possam acompanhar a evolução histórica da reputação de um determinado usuário.

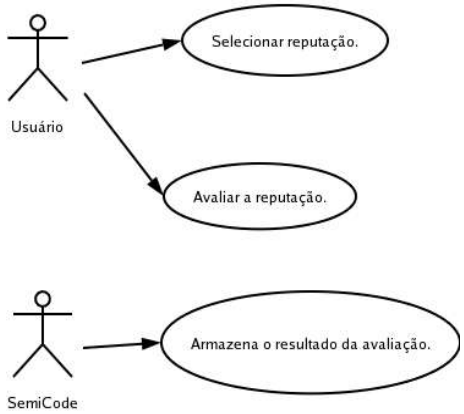


Figura 3.6 – Pacote de Casos de Uso: Usuário interage com a reputação do par selecionado.

Outra possibilidade de interação com um par remoto é o acesso às

informações de compartilhamento de arquivos desse par, que pode ser visualizada na figura 3.7. Dessa forma, o usuário pode selecionar um par e visualizar quais são os arquivos que esse usuário disponibiliza para acesso na rede. A partir disto, o usuário pode selecionar um arquivo desejado e fazer o *download* do mesmo para sua máquina local.

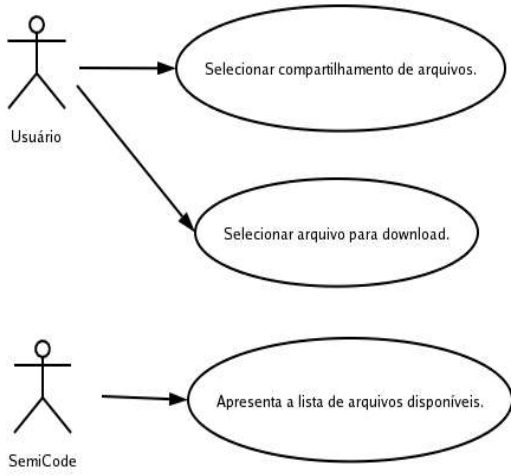


Figura 3.7 – Pacote de Casos de Uso: Usuário interage com o compartilhamento de arquivos do par selecionado.

Além dos módulos acima descritos, o SemiCode permite ao usuário do sistema utilizar um mecanismo de envio de perguntas e respostas aos outros pares presentes na rede, que pode ser visualizado na figura 3.8. Nesse mecanismo o SemiCode armazena o histórico da sessão atual de uso para que o usuário possa visualizar quais as questões e respostas foram trocadas nessa determinada sessão.

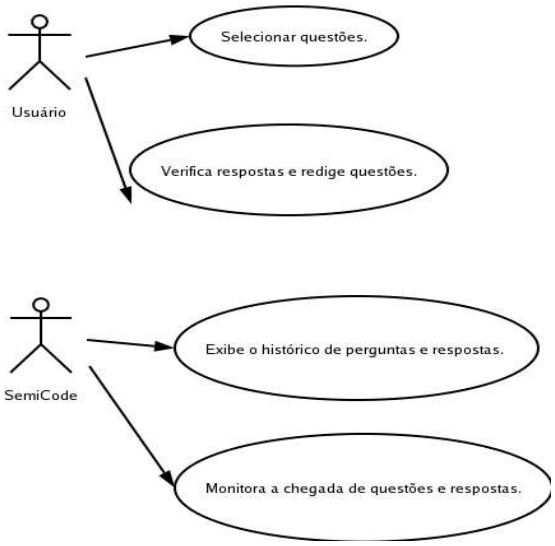


Figura 3.8 – Pacote de Casos de Uso: Usuário interage das questões com o par selecionado.

3.2.3 Diagrama de Classes

O diagrama de classes consiste basicamente de um diagrama estático que representa as relações entre as classes do sistema sem considerar o fator temporal e que busca representar as abstrações reais que são evidenciadas quando desenvolvemos o domínio do problema.

Analisando os textos acima apresentados do domínio do problema e dos casos de uso, torna-se de fácil percepção quais são os elementos que constituem o núcleo do sistema SemiCode e, portanto, quais são os elementos dessa modelagem que iremos utilizar como classes na construção desse diagrama de classes.

Um ponto fundamental que aqui cabe análise é a questão da modularização das funcionalidades de interação do sistema. A idéia é que se num dado momento tivermos a idéia de substituir o sistema de *chat* clássico por troca de mensagens de texto para um sistema de videoconferência, por exemplo, isso tenha um mínimo impacto no código do sistema. Essa característica permite também que desenvolvedores externos possam desenvolver módulos para o sistema utilizando apenas como referência uma interface comum e que não se altera ao longo do projeto e da implementação de software.

Dessa forma, podemos ressaltar aqui quais foram os elementos principais que foram extraídos da descrição do sistema e dos casos de uso. Temos os módulos de interação do sistema, a saber: chat, arquivos, reputação e questões. Aliado ao módulo de reputação temos o perfil do usuário, que serve como base de dados a respeito das preferências pessoais de um determinado usuário. Para todos esses módulos temos características comuns de implementação que podem ser captadas por uma interface, o que permite separar a implementação dos módulos do conceito de um módulo SemiCode, facilitando dessa forma a manutenção do sistema e o desenvolvimento e atualização de módulos.

Além dos módulos relatados acima, temos uma série de operações de monitoramento, envio, recepção e o tratamento de mensagens que trafegam pela rede P2P. Essas tarefas devem ser implementadas por uma classe que funciona como uma espécie de núcleo por onde todas as mensagens circulam e são encaminhadas.

Para finalizar os módulos principais da modelagem de classes, temos ainda a interface com os protocolos da especificação JXTA que permitem enviarmos e recuperarmos informações da rede P2P, que é representada pelas classe EZMinimalPeer, que é uma classe que implementa todas as funcionalidades mínimas para conexão entre pares no sistema.

Podemos visualizar na figura 3.9 a modelagem proposta.

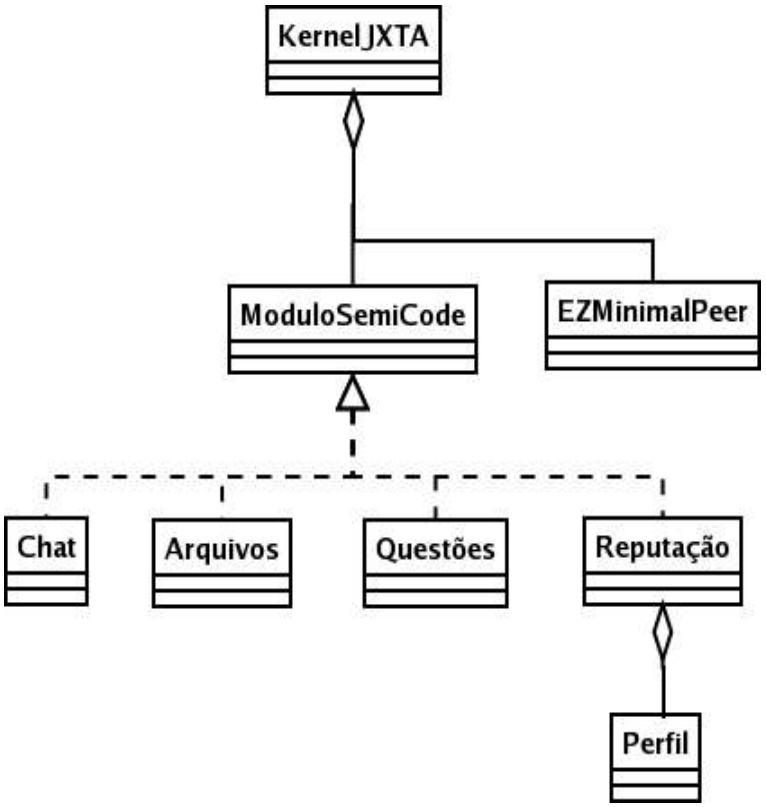


Figura 3.9 – Diagrama de Classes do SemiCode.

3.3 Resumo

Neste capítulo, desenvolvemos a especificação conceitual do SemiCode com a descrição de cada um dos módulos que irão compor o sistema, bem como das funcionalidades agregadas. Essa especificação representa uma primeira reflexão a respeito da interface que se deseja entre os conceitos de uma rede descentralizada formada por pares e a tecnologia de rede estudada nos capítulos anteriores. Essa interface se desdobra num segundo momento nas especificações do sistema baseadas na metodologia ICONIX, que nos permitiu elaborar uma descrição do domínio do problema a ser resolvido. É a partir dessa descrição que pudemos construir os casos de uso e assim mapear a relação do usuário do sistema com as funcionalidades que foram pensadas na descrição conceitual do SemiCode.

Por final, apresentamos o diagrama de classes estático do sistema, onde pudemos analisar o relacionamento entre as classes e algumas questões

conceituais de funcionamento e organização do sistema que irão ter seu desdobramento na implementação e no uso cotidiano do SemiCode.

É esse estudo e a modelagem relacionada ao mesmo que nos permitiu partir para a implementação do sistema e buscar as soluções tecnológicas para a resolução dos problemas encontrados no processo de desenvolvimento, parte esta que é relatada no próximo capítulo.

Capítulo 4 – Tecnologia e Implementação do SemiCode

Após a modelagem e conceituação dos módulos que compõem o sistema SemiCode, iremos nesse capítulo descrever a tecnologia que foi empregada na implementação e apresentar o sistema na forma como ele foi concebido. Trataremos também de algumas questões específicas de implementação, como estruturas de dados utilizadas e o sistema de monitoramento de informações na rede, o que nos permite descrever os elementos de fundamental importância para o funcionamento do SemiCode.

Dessa forma, este capítulo está organizado de forma a apresentarmos inicialmente a interface JAL (*Jxta Abstraction Layer*) http://ezel.jxta.org/jal.html) que utilizamos para a programação com os protocolos e a implementação do JXTA, as estruturas de dados utilizadas para a organização da informação no sistema, o mecanismo de monitoramento da rede e uma apresentação da implementação do SemiCode.

4.1 A interface JAL

A idéia por trás da criação do projeto JAL era oferecer à comunidade de desenvolvedores de software em P2P uma abstração em relação às funcionalidades do mundo P2P e tornar essa tecnologia de uma certa forma mais simples e acessível, de forma a que a interface pudesse ser um meio de comunicação com diferentes tipos de implementações de protocolos P2P. Portanto, seu desenvolvimento visa principalmente tornar mais simples o acesso às funcionalidades principais das implementações do JXTA e outras possíveis que possam ser conectadas pela interface JAL.

Basicamente, o JAL é uma API que tem a intenção de ser imutável para as primitivas mais usadas em P2P, independente da implementação que estamos usando. Esse fato permitiria, por exemplo, que num determinado momento pudéssemos trocar a implementação do JXTA para uma outra tecnologia que fosse de maior interesse ou apresentasse melhor performance para o tipo de projeto que estamos desenvolvendo. Dessa forma, não temos nenhuma necessidade de alterar nosso código para usar uma outra tecnologia ou

especificação de protocolos P2P. Portanto, a escolha para usarmos o JAL na implementação do SemiCode consiste dessas duas razões principais, sendo a primeira a facilidade de acesso as primitivas da implementação JAVA do JXTA e a segunda a API para mundo da programação P2P que permitiria uma futura atualização nos protocolos utilizados no projeto sem a necessidade de alteração alguma em nosso código. Podemos observar na figura 4.1 essas características do JAL.

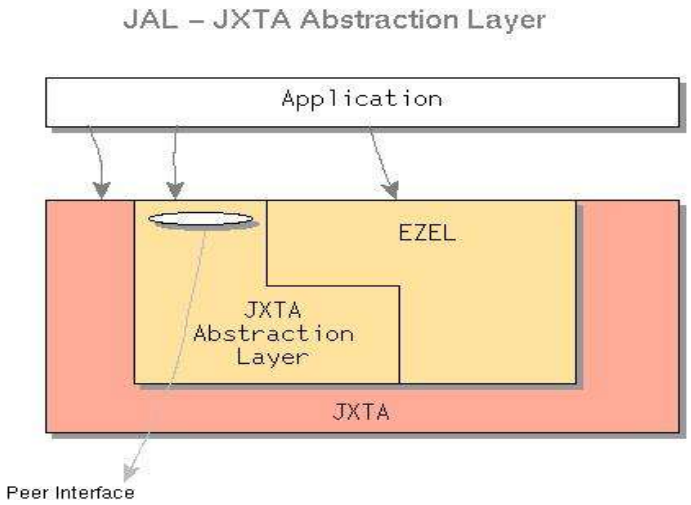


Figura 4.1 – A interface JAL (fonte: http://ezel.jxta.org/jal.html).

Os objetivos principais do JAL consistem em permitir que o usuário da interface possa implementar ou utilizar a implementação padrão para atividades básicas relacionadas a descobrir pares numa rede e permitir ser descoberto por outros pares, procurar por outros pares e grupos de pares, criar e gerenciar um grupo de pares, comunicação com outros pares e obter informações sobre outros pares e grupos de pares.

A implementação da interface fica a cargo do desenvolvedor; no entanto, o projeto oferece uma implementação padrão focada na implementação JAVA do JXTA, que é a classe EZMinimalPeer (presente no diagrama de classes da figura 3.9 do capítulo 3) que implementa a interface Peer, que pode ser vista na figura 4.1. Essa interface Peer é a responsável por podermos utilizar as atividades básicas acima relacionadas. Foi essa implementação padrão que utilizamos no desenvolvimento do SemiCode.

Podemos colocar aqui um pequeno exemplo conceitual que permite

visualizar o funcionamento do JAL e a forma como ele pode ser utilizado para a construção de programas que utilizam primitivas de redes P2P. Um código exemplificando o diagrama da figura 4.2 pode ser encontrado no código 1 do apêndice A.

Figura 4.2 – Criação e inicialização de um par via JAL.

Há aqui três atividades básicas que são oferecidas pela interface do JAL de forma a permitir a criação de um grupo de pares e a junção do par que o criou a esse mesmo grupo. A rede é inicializada pela interface de forma a permitir a criação de um grupo de pares, o que é feito logo em seguida. De forma a permitir a troca de mensagens no âmbito desse grupo, o par que o criou junta-se ao mesmo e a partir daí temos que qualquer outro par que se inicializar no mesmo grupo fará parte da mesma comunidade.

Dessa forma, podemos ter uma noção geral do uso da interface JAL para primitivas básicas de uso comum em redes P2P. O sistema SemiCode utiliza o mesmo processo para inicializar os pares na rede, criar um grupo e obter informações de pares e grupos de pares. Após analisarmos a forma como o SemiCode se relaciona com as primitivas da rede P2P, podemos passar agora a análise das principais estruturas de dados utilizadas na implementação de forma a suportar as atividades de interação do sistema.

4.2 Estrutura de Dados

O SemiCode utiliza basicamente duas estruturas de dados que permitem a

realização das atividades do sistema e do suporte a informação utilizada. Aqui temos as estruturas de arquivos, onde informações como o perfil do usuário e sua reputação relacionada ficam armazenados e as estruturas dinâmicas em memória que permitem captarmos todas as informações que são recebidas por um determinado par e que circulam pela rede P2P.

A estrutura de arquivos do SemiCode trabalha com o conceito da serialização de objetos. Através dessa serialização, podemos armazenar em arquivos as informações registradas pelo usuário através de um objeto instanciado para tal fim. Dessa forma, podemos aqui armazenar os dados de perfil de um determinado usuário e registrar sua reputação através das avaliação de seus pares. Devido ao fato dos dados serem armazenados em objetos, facilita o envio e o recebimento dos mesmos através das mensagens de comunicação em rede entre pares. Para obter maiores informações a respeito de como esse processo foi implementado no SemiCode, colocamos propostas de implementações no apêndice A, nos códigos 2 a 4.

As estruturas dinâmicas que utilizamos na implementação do SemiCode visam basicamente servir como um repositório de informações onde podemos armazenar as mensagens que chegam através da rede P2P e os respectivos remetentes. Dessa forma, criamos condições de permitir que uma fase de pré-triagem das mensagens possa armazenar cada tipo de mensagem em sua respectiva estrutura de dados, o que permite uma atualização constante das interfaces gráficas com o usuário e o contínuo monitoramento da rede, que será visto em detalhes na seção 4.3.

Quando falamos em tipos de mensagens, é importante relatar que o SemiCode diferencia por tipo de interação cada mensagem que é enviada na rede. Isto quer dizer que mensagens do tipo chat, troca de arquivos, questões e mesmo reputação possuem um cabeçalho ligeiramente diferente, o que permite que identifiquemos qual o tipo de mensagem que um determinado usuário recebeu.

A estrutura de dados que permite que façamos esse tipo de armazenamento da informação não pode ser estática, ou seja, não pode ter um número fixo de posições de memória reservado e sim poder alocar posições conforme a necessidade de uso. Dessa forma, utilizamos as estruturas de dados

fornecidas pelo JAVA, que funcionam como uma listas ligadas onde podemos ir alocando mais posições conforme precisamos armazenar mais informações. Conceitualmente, temos na figura 4.3 as atividades envolvidas nesse processo.

Figura 4.3 – Analise de mensagens do SemiCode.

Inicialmente, temos que o sistema irá verificar o cabeçalho de toda a mensagem que for recebida por um determinado par. Essa verificação irá identificar a que tipo de interação a mensagem está relacionada, ou seja, chat, arquivos, reputação, etc. Após essa etapa, a mensagem é armazenada em sua estrutura adequada, o que permitirá a cada serviço identificar uma nova mensagem de interação e apresentá-la ao usuário em sua respectiva interface gráfica. Ao armazenar, o sistema irá verificar se o serviço já está inicializado, o que efetivamente significa que o usuário já se encontra em interação com esse tipo de serviço com o par responsável pelo envio dessa mensagem. Se o serviço já estiver inicializado, o SemiCode apenas irá aguardar que ele recupere a mensagem e a informe ao usuário; se ainda não estiver inicializado, uma interface gráfica com o usuário será apresentada informando da nova interação em processo.

Podemos analisar como o SemiCode implementa esse tipo de estrutura no apêndice A código 5.

4.3 Monitoramento da Rede

A questão do monitoramento da rede está relacionada com a leitura constante de mensagens que um determinado usuário do sistema recebe de seus pares e com a atualização das interfaces gráficas com o usuário. Essa última questão apresenta um problema de implementação a ser resolvido quando usamos o ***Swing, opção adotada no desenvolvimento do SemiCode***, que é um *framework* java (**javax.swing**) que permite compor diversos componentes, como caixas de texto e botões, na interface com o usuário. Esse problema é relativo a termos uma instância do sistema que fica "escutando" a rede para verificar se informações novas chegaram como, por exemplo, a saída de um par da rede ou a chegada de um novo usuário e a necessidade de atualizar a interface gráfica com o usuário conforme os dados que foram detectados nesse processo de escutar a rede. O problema fica ainda mais evidente quando temos uma troca síncrona de mensagens entre usuários, o que é o caso do chat. Precisamos enviar e receber constantemente informações da rede e atualizar o mais rápido possível a interface gráfica para permitir uma dinâmica real de conversação entre dois pares.

A solução para esse problema consiste na implementação da programação com várias linhas de execução simultâneas (multithreading). Basicamente, aqui implementamos dois processos que ocorrem de forma concorrente na máquina do usuário do sistema, ou seja, um processo é o responsável por atualizar a interface gráfica com novas mensagens e informações do sistema e outro processo é responsável por escutar a rede. Cada um tem uma determinada fatia de tempo do processador, o que evita o efeito de espera de execução de um processo para que outro possa ser executado, o que certamente causaria o travamento da interface gráfica com o usuário.

Detalhes da implementação podem ser encontrados no apêndice A, código 6.

É este, portanto, o mecanismo de monitoramento de mensagens que permite a atualização contínua da interface gráfica com o usuário com informações relevantes dos processos de interação do SemiCode.

Passaremos agora à apresentação do SemiCode da forma como ele foi implementado.

4.4 O SemiCode

Iremos agora apresentar o SemiCode através de suas interfaces com o usuário e suas atividades de interação

4.4.1 Autenticação no sistema

Inicialmente, como apresentado na figura 4.4, temos a autenticação do usuário no sistema.

Figura 4.4 – Registro do nome do par representado pelo usuário no SemiCode.

Após esse registro inicial, o usuário do sistema ainda passa por uma segunda autenticação, essa já relacionada a sua instalação do JXTA em sua máquina local, visto na figura 4.5.

Figura 4.5 – Autenticação do usuário na rede JXTA.

4.4.2. Perfil do usuário

Realizadas as operações de autenticação no sistema, o SemiCode irá agora analisar se o usuário já tem seu perfil preenchido e registrado em seu sistema de arquivos. É através desse perfil que um usuário pode ser conhecido na rede e informar questões relativas as suas especializações no trabalho, preferências pessoais, entre outras informações que podem auxiliar no processo de conexão e

colaboração entre os pares. Se esse perfil não for identificado pelo sistema, o SemiCode irá antes de inicializar a interface de interação entre pares levar o usuário ao preenchimento de seu perfil.

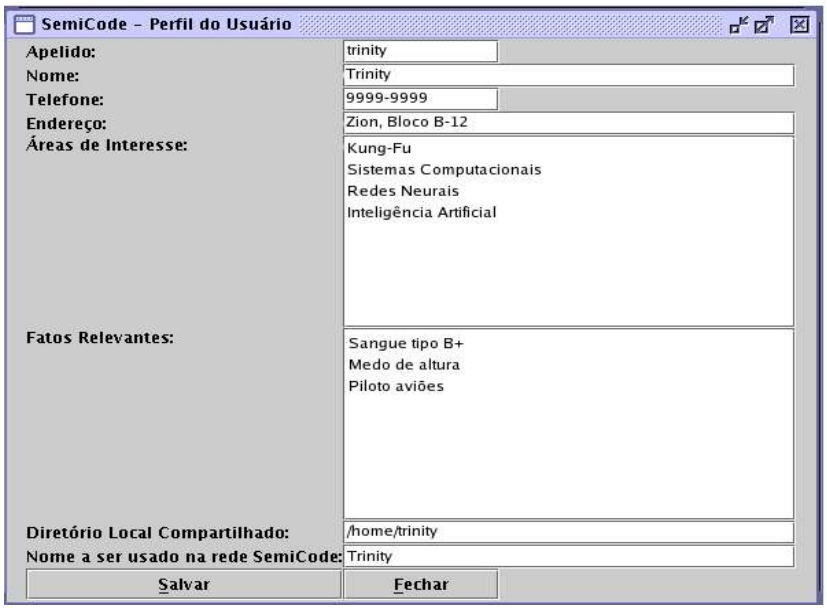


Figura 4.6 – Tela de registro do perfil do usuário no SemiCode.

É importante notar que na figura 4.6 o usuário do sistema também deve preencher o diretório local que ele deseja compartilhar na rede. É a partir desse diretório que os outros pares conectados na rede poderão compartilhar seus arquivos e escolher arquivo de outros pares para baixar em sua máquina local.

4.4.3. Interface principal

A partir desse momento, o SemiCode está preparado para apresentar ao usuário do sistema sua interface de interação, na figura 4.7, que permitirá esse usuário observar outros pares na rede e iniciar ações de interação com os mesmos.

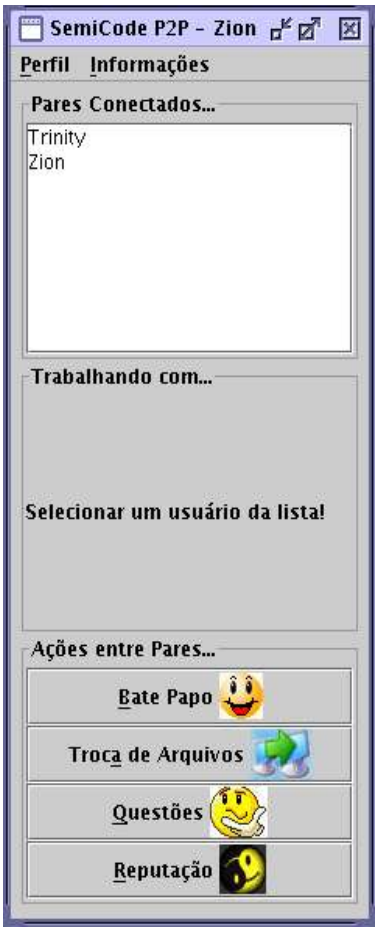


Figura 4.7 – Interface de interação com o usuário e a rede JXTA.

Nesse ponto, vale a pena notarmos algumas características específicas dessa interface que estão relacionadas à forma como o sistema opera a colaboração entre pares na rede. O primeiro ponto importante a ser destacado é que temos quatro áreas nessa interface com o usuário, a área de menu que é onde podemos obter mais informações sobre o sistema e visualizar ou alterar o registro de perfil do usuário, a área de "pares conectados" que é onde podemos visualizar quais são os pares que estão ativos no sistema nesse momento (essa área é constantemente atualizada, portanto indica a saída de pares e a entrada de novos pares), a área "trabalhando com" que representa o par que selecionamos a partir da lista de pares conectados para interação e, por final, a área "ação entre

pares" que permite selecionarmos entre as opções de interação que estão implementadas nessa versão do SemiCode. Outro ponto importante a ser destacado é a forma como escolhemos um par para interagir da lista de pares conectados. Esse processo ocorre através do *drag'n'drop* entre o nome de um par na lista de pares conectados e área da interface chamada "trabalhando com", que pode ser visto na figura 4.8.

Figura 4.8 – Selecionando um nome da lista de pares conectados para interação.

A figura 4.8 demonstra a selação de um nome, no caso Trinity, e seu aparecimento na área "trabalhando com". Esse processo tem por objetivo levar o usuário a escolher um par na rede, ação que está diretamente relacionada com as possibilidades de interação entre pares. É uma forma de dar um destaque ao par com o qual o usuário está trabalhando e permitir que o mesmo tenha preferência de interação nessa determinada seção de uso do sistema. Essa ação de selecionar um determinado par na rede também habilita a

possibilidade de a partir desse ponto utilizar os botões de ações entre pares, que efetivamente vão permitir os processos de troca de informações entre os pares.

4.4.4. Chat

Agora, podemos iniciar nossa análise dos módulos de interação entre pares, começando pelo *chat* entre pares, que é apresentado na figura 4.9.

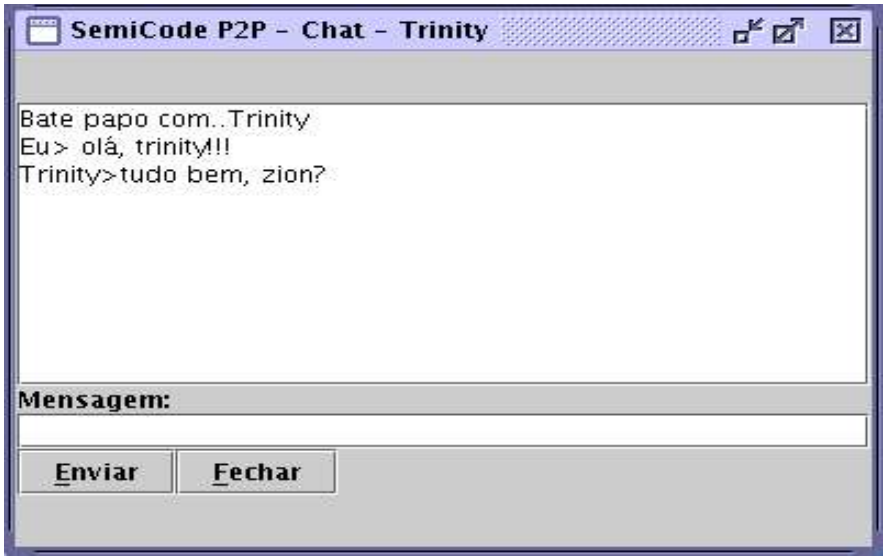


Figura 4.9 – Interface de *chat* entre pares.

Após o usuário ter selecionado um par e iniciar uma ação de *chat* entre ambos, o SemiCode apresenta a interface de interação da figura 4.9. Nessa tela, podemos destacar alguns elementos. Já no título da janela, o sistema apresenta o nome do par com o qual o usuário estabelece uma conversação, característica que pode auxiliar a encontrar uma determinada janela num cenário em que temos muitas conversas ocorrendo ao mesmo tempo. A informação de com quem o usuário conversa aparece novamente no início da área de apresentação de texto para o usuário. Após isto, qualquer mensagem que o usuário do sistema digitar no campo "Mensagem" e pressionar o botão "Enviar" será apresentanda no campo de texto com a etiqueta "Eu>" antes do texto, de forma a diferenciar das mensagens que

estão sendo enviadas pelo par remoto e que aparecem sempre com o nome do par que as enviou antes. Essa interface permite troca indefinida de mensagens entre os usuários desde que ambos permaneçam conectados no sistema, pois o campo de texto possui barras de rolagem permitindo a chegada contínua de novas mensagens.

4.4.5. Arquivos

A próxima ação de interação que um usuário do sistema pode escolher é a troca de arquivos entre pares, que é apresentada na figura 4.10.

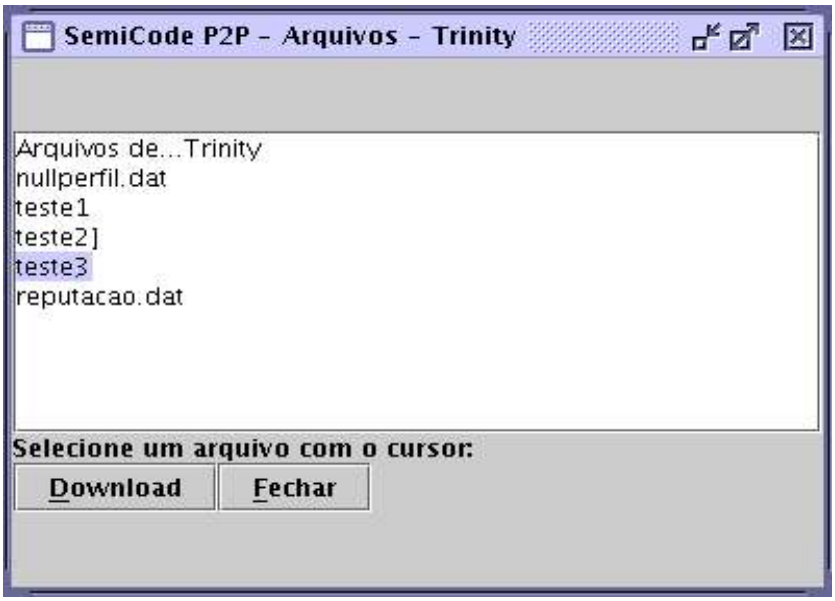


Figura 4.10 – Interface de troca de arquivos entre pares.

Podemos aqui destacar alguns pontos relevantes da interface da figura 4.10. Incialmente, temos as mesmas características da interface de *chat* preservada, que é basicamente o nome do sistema no título da janela seguido da opção de interação e do nome do par com o qual um usuário está trabalhando nesse momento. A seguir a informação de quem pertencem os arquivos é repetida na primeira linha da janela de texto e, após, os arquivos que o par remoto disponibilizou em seu diretório de arquivos compartilhados. Apresentada essa listagem de arquivos, o usuário do sistema pode

resolver apenas fechar a interface ou baixar um arquivo em seu diretório local de compartilhamento. O processo de baixar um arquivo consiste basicamente de selecionar um arquivo através do *mouse* e clicar no botão "*Download*". Esse processo fará com que o SemiCode envie uma mensagem para o par remoto requisitando o arquivo selecionado e o par irá então responder com uma mensagem cujo conteúdo será o arquivo selecionado. Após o usuário receber essa mensagem com o arquivo como conteúdo, o sistema irá armazenar as informações no diretório local do usuário que está compartilhado com o resto dos pares. Dessa forma, o SemiCode permite que um recurso disponibilizado na rede possa ser multiplicado e dessa forma possuir vários pares que o armazenam e disponibilizam, o que se torna interessante num cenário de rede P2P, onde nem sempre temos todos os pares conectados ao mesmo tempo.

4.4.6. Questões

Passemos agora a análise do módulo de questões, que é apresentado na figura 4.11.

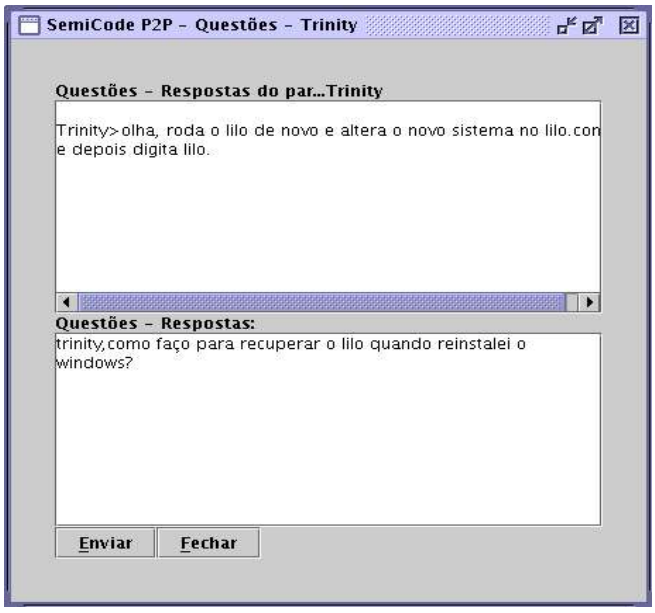


Figura 4.11 – Interface de questões entre pares.

Nesse módulo, temos duas áreas na interface gráfica. A primeira área de texto na interface representa a área onde o usuário do SemiCode irá receber as respostas para as questões que ele enviou. A segunda área é onde ele pode escrever suas perguntas e enviar para o par remoto. Um fator importante a ser mencionado neste módulo é o fato de que ele guarda o histórico das respostas que um determinado par remoto enviou ao usuário do sistema. Isso representa que durante a sessão de uso do sistema, toda vez que o usuário quiser enviar ou for receber uma questão desse mesmo par a interface gráfica irá exibir as respostas e as questões já recebidas. Essa é uma forma de permitir que o usuário sempre tenha acesso a essas informações que são menos coloquiais e mais focadas numa dinâmica assíncrona de conversação do que o módulo de *chat*.

4.4.7. Reputação

Passemos agora a análise do módulo de reputação, que é apresentado na figura 4.12.

Figura 4.12 – Interface de reputação entre pares.

Podemos destacar nesse módulo de reputação a questão de utilizarmos um sistema de forma experimental, que permita introduzirmos essa

questão de forma prática no uso e na análise de sistemas colaborativos. Inicialmente, o sistema apresenta para o usuário do SemiCode o perfil que o par remoto tem de si mesmo em sua máquina local. Exibir esse perfil tem por objetivo apresentar de uma certa forma o par ao usuário que deseja obter maiores informações a seu respeito. Logo após as informações do perfil do par, aparecem três linhas, colaborativo, presente e ausente e uma avaliação em porcentagem ao lado das mesmas. Essas linhas simbolizam a avaliação que esse par tem tido na rede. Essa avaliação é sempre absoluta em relação a quantidade de votos que esse par tem tido na rede, por isso o uso da porcentagem. Logo abaixo, é apresentado ao usuário do sistema três opções de avaliação desse par. O usuário pode escolher qualquer um dos itens e salvar sua opção de avaliação do par. Esse processo de salvar consiste em enviar uma mensagen de avaliação ao par remoto e o mesmo irá salvar essas informações em seu arquivo local que armazena seus dados de reputação. Ao salvar esses dados em arquivo, as porcentagens serão atualizadas e da próxima vez que um outro usuário resolver obter as informações de reputação desse par os dados já estarão atualizados.

Esse formato permite termos uma evolução no tempo da avaliação dos pares no sistema, o que pode auxiliar na questão de os usuários irem conhecendo quais são os pares que mais colaboram na rede, respondem mais questões e auxiliam diretamente no andamento dos trabalhos colaborativos.

4.4.8. Informações

Por fim, iremos apresentar a opção do menu principal que oferece maiores informações sobre o SemiCode na figura 4.13. Essa janela tem por objetivo fornecer informações a respeito da origem do sistema e formas de contato.

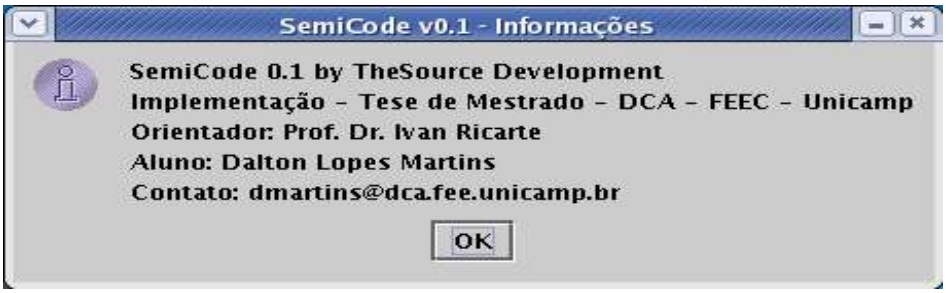


Figura 4.13 – Informações sobre o SemiCode.

4.5 Avaliação do SemiCode

Analisando a implementação do SemiCode da forma como foi proposta podemos destacar uma série de características que nos levam a refletir sobre o sistema e seu formato de interação.

Um ponto básico e de fundamental importância é o formato desenhado para a interface gráfica com o usuário. Buscou-se ter uma relação bastante direta e de fácil compreensão, que pudesse resgatar as metáforas utilizadas pelos programas de mensagem instantânea mais comuns disponíveis nas várias plataformas. Além disso, o ponto chave da interface é o encontro do par, o que tem bastante destaque na seção Pares Conectados da interface com o usuário, que leva o mesmo a efetivamente encontrar um par disponível em sua lista antes de iniciar qualquer atividade de interação e colaboração em rede. Outro destaque efetivo e que surgiu da necessidade de se evidenciar a interação entre usuários é a questão do par com o qual estamos trabalhando num dado momento de uso do sistema, sendo que o mesmo tem uma seção da tela somente para destaque de seu nome. Essa característica trouxe como consequência a viabilização das atividades de interação após a seleção do par.

Certamente, caberiam aqui outras metáforas que pudessem ser utilizadas como forma de capturar a atenção do usuário e de suas iniciativas de interação. Uma das alternativas seria integrar todas as atividades de interação em uma única janela de interface com o usuário, mas, por desejarmos permitir ao usuário a modularização completa de suas atividades de interação, deixamos que as mesmas pudessem ser executadas em janelas

separadas.

A troca de arquivos proposta pelo sistema buscou implementar o conceito do compartilhamento efetivo não apenas de arquivos mas de espaço em disco específico dos pares conectados na rede. Um usuário do sistema tem acesso diretamente ao diretório disponibilizado para compartilhamento por um outro par, o que o permite verificar em quais trabalhos esse par está envolvido, como evoluem seus projetos e como ele mesmo pode utilizar aquele material como base de seus trabalhos ou complemento de atividades em andamento. É, portanto, uma forma de colaboração sem a necessidade de que um par na rede precise gastar mais tempo do que simplesmente gravar um arquivo em seu diretório compartilhado.

Já o sistema de questões nos permitiu criar um histórico de troca de informações de um período de uso do sistema em andamento, o que se mostra útil na execução de atividades detalhadas, onde as informações precisam ser consultadas constante pelos usuários do sistema.

Outro ponto importante e ainda emergente como questão a ser debatida e experimentada em sistemas computacionais é o caso da reputação do usuário em relação ao julgamento de seus pares. Procuramos nesse ponto apresentar uma solução simples e que permitisse gerar uma métrica de fácil análise e compreensão por parte dos usuários. É um ponto ainda a ser explorado e analisado ao longo de sua evolução em outras implementações.

Sobre a implementação em termos do código desenvolvido, teríamos a alternativa de não usarmos a interface JAL e programarmos diretamente em JXTA. No entanto, como o JAL é um projeto desenvolvido pela própria comunidade JXTA e que tem por objetivo ser uma interface de uso geral para P2P, resolvemos utilizar a interface de forma a contribuir com seu desenvolvimento e aplicação de forma geral.

Foi portanto este formato de relacionamento com o usuário e com o sistema como um todo que encontramos para mapear todas as necessidades e questões levantadas nos primeiros capítulos deste trabalho, em relação aos hábitos e formas como os usuários se relacionam em trabalhos colaborativos e em rede.

4.6 Resumo

Este capítulo teve por objetivo apresentar as tecnologias que foram empregadas no desenvolvimento do SemiCode e sua implementação propriamente dita. Dessa forma, pudemos organizar a apresentação das informações de forma a privilegiar a maneira como o sistema foi sendo desenvolvido. Inicialmente, realizamos os testes e o estudo da tecnologia de conexão com a implementação do JXTA para o mundo JAVA, que se deu através do JAL. Após utilizarmos e dominarmos o processo de programação com o JAL, passamos a desenvolver e especificar as estruturas de dados que iriam dar suporte às atividades de interação propostas na especificação do sistema. Essas estruturas estavam divididas nas estruturas dinâmicas para dar suporte apenas a uma sessão de uso do sistema, portanto não precisavam armazenar informações para sessões posteriores e estruturas de acesso a arquivos em disco, que permitiam armazenar informações que seriam utilizadas em outras sessões de uso do SemiCode. Resolvidos esses problemas, pudemos nos concentrar nas estruturas que permitiriam o monitoramento efetivo da rede P2P e ainda assim a constante atualização das informações na interface gráfica com o usuário, o que se deu na utilização da programação *multithread.*

Apresentadas essas questões, portanto, pudemos ao final apresentar o sistema SemiCode como foi implementado e suas interfaces de interação com o usuário, bem como uma pequena avaliação do sistema em relação a algumas expectativas levantadas nos primeiros capítulos deste trabalho. No próximo capítulo iremos apresentar uma sessão de uso do SemiCode em comparação com o uso de sistema tradicional baseado na arquitetura cliente/servidor e uma proposta híbrida P2P/Cliente–Servidor.

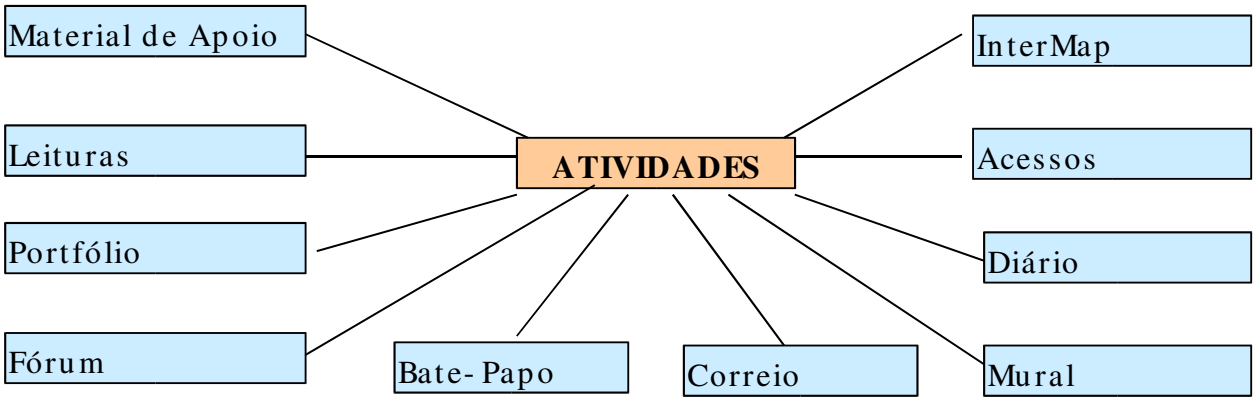
Capítulo 5 – O SemiCode e a arquitetura cliente/servidor

Durante o processo de pesquisa e estudo que se relacionaram ao desenvolvimento do sistema SemiCode, sistemas baseados na arquitetura cliente/servidor sempre foram considerados como referência de funcionalidades e pontos de análise para a reflexão de como a tecnologia *peer-to-peer* poderia oferecer novos paradigmas do suporte à colaboração em rede. Portanto, decidimos dedicar um capítulo desta tese para uma proposta onde poderemos ter um sistema P2P e cliente/servidor atuando de forma a se complementarem. O sistema cliente/servidor estudado é o sistema TelEduc[1] desenvolvido pelo Núcleo de Informática Aplicada à Educação (NIED) da Unicamp.

Portanto, este capítulo está organizado de forma a apresentarmos inicialmente o sistema TelEduc e, após esta apresentação inicial, uma proposta onde o sistema SemiCode e a arquitetura P2P podem servir como um complemento aos sistemas cliente/servidor, como o TelEduc.

5.1 O TelEduc

O TelEduc é um ambiente focado no que podemos chamar de ensino a distância (EaD) que foi concebido inicialmente para auxiliar no processo de formação de professores para a Informática na Educação (Vieira, 2002). Podemos analisar sua arquitetura básica através da figura 5.1.



1 Software livre disponível para download em http://teleduc.nied.unicamp.br

Figura 5.1 – Arquitetura Básica do ambiente TelEduc – (Otsuka, Vieira, 2002).

Analisando essa arquitetura básica, podemos notar que o sistema está fortemente baseado no desenvolvimento de atividades, ou seja, na realização de tarefas determinadas que contarão com o apoio de outras ferramentas no processo de desenvolvimento do trabalho. Segundo Vieira (2002), "isto vem ao encontro do pressuposto de que o aprendizado de conceitos de qualquer domínio do conhecimento é feito a partir da resolução de problemas, com o subsídio de diferentes materiais como textos, *software* e instruções de uso que podem ser colocados para o aluno por meio de ferramentas como: material de apoio, leituras, perguntas freqüentes *etc*."

Dessa forma, todas as ferramentas implementadas no sistema têm como foco dar suporte à realização de atividades num ambiente de um curso, uma disciplina escolar onde essas atividades são determinadas por um professor, tutor ou monitor da disciplina com o objetivo de servirem como registro do processo de aprendizagem e documentação de suporte à avaliação do desenvolvimento e conhecimento adquirido por um aluno que interage com as informações disponibilizadas pelo ambiente.

Para a completa compreensão do TelEduc temos que o sistema é dividido em três grupos de ferramentas: ferramentas de coordenação, ferramentas de comunicação e ferramentas de administração.

5.1.1 Ferramentas de coordenação

As ferramentas de coordenação são aquelas diretamente ligadas a dinâmica de funcionamento de um curso e temos nesse conjunto a ferramenta Atividades, responsável por indicar ao aluno o que ele deve realizar num determinado intervalo de tempo em relação ao decorrer natural do curso ao qual ele está inscrito. Podemos visualizar a ferramenta Atividades na figura 5.2. Vale ressaltar aqui que essa ferramenta está diretamente relacionada a dinâmica de funcionamento de um curso, pois é ela que vai ditar o ritmo de trabalho de um estudante, o tipo de atividades que ele tem de realizar e a

própria forma como o mesmo deve se relacionar através do ambiente de forma a obter as informações necessárias para a realização das atividades propostas.

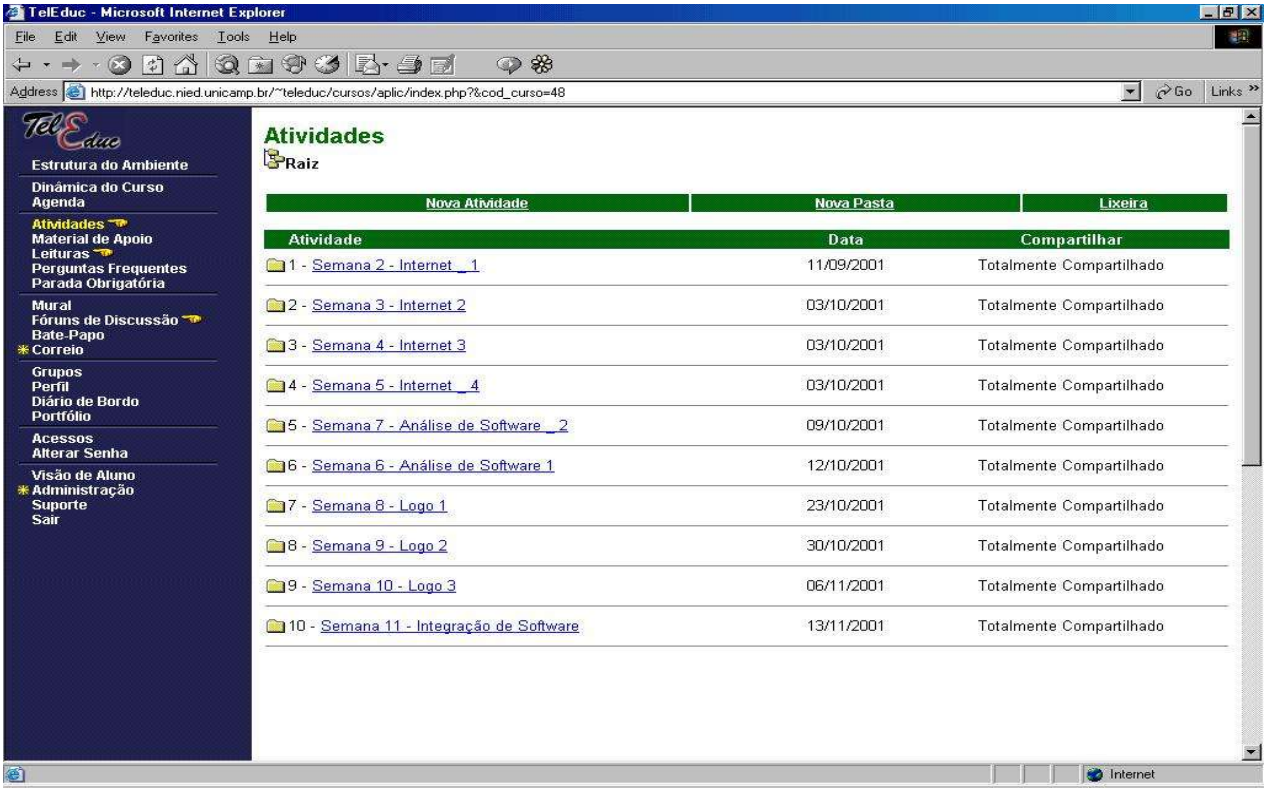


Figura 5.2 – Atividades propostas pelo sistema – (Otsuka, Vieira, 2002).

Na figura 5.2, podemos notar o sistema de disponibilização das atividades oferecidas pelo sistema. Elas estão basicamente organizadas de forma temporal, ou seja, atividades a serem desenvolvidas por semana que devem ser seguidas pelos estudantes cadastrados no ambiente para a realização de um determinado curso disponibilizado. É, portanto, em torno dessa disponibilização de atividades a serem executadas que as outras ferramentas do sistema se relacionam entre si.

Também do conjunto de ferramentas de coordenação temos s ferramentas de Material de Apoio e Leituras que são utilizadas como referências às informações utilizadas pelos

estudantes durante a realização de atividades propostas pelo sistema. A ferramenta Material de Apoio está diretamente vinculada a informações a respeito de uma atividade específica e a ferramenta Leituras é usada como forma de oferecer textos e referências de âmbito geral em relação aos tópicos estudados no curso em questão.

5.1.2 Ferramentas de comunição

Passaremos agora à análise das ferramentas de comunicação disponibilizadas pelo TelEduc. Nesse conjunto temos as ferramentas Portfólio, Fórum, Bate-papo, Correio, Diário e Mural.

A ferramenta Portfólio (figura 5.3) pode ser vista como uma área de disco no servidor onde o TelEduc está instalado, disponibilizada ao estudante cadastrado no sistema onde o mesmo pode realizar o *upload* de arquivos que desejar. Essa é uma forma que o estudante possui de disponibilizar o material que tem produzido para que outros estudantes e o tutor do curso possam visualizar esse material e desenvolver comentários a respeito ou mesmo avaliá-lo.

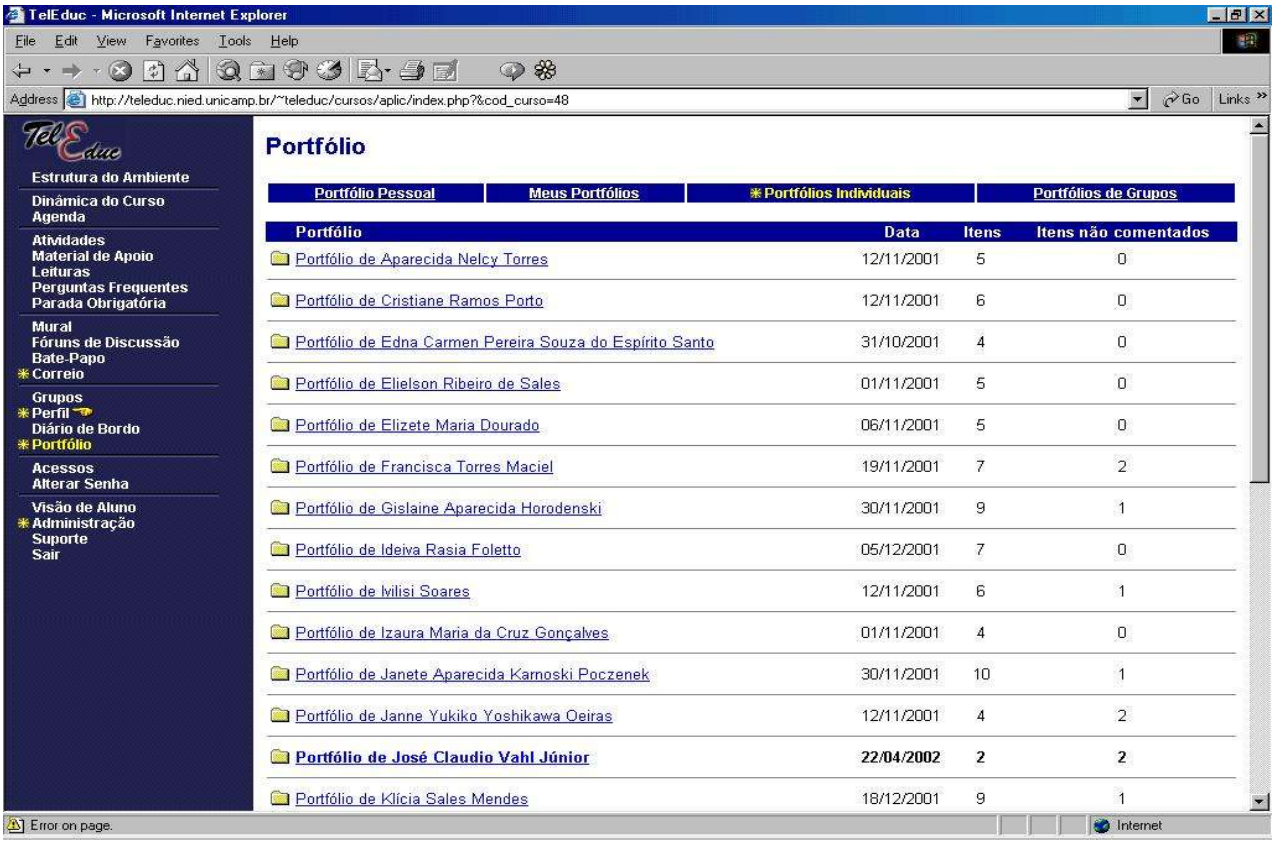


Figura 5.3 – Ferramenta Portfólio – (Otsuka, Vieira, 2002).

As ferramentas Fórum, Bate-papo e Correio são implementadas de forma tradicional, ou seja, como normalmente podemos encontrar esse tipo de ferramenta disponibilizada na Internet.

A ferramenta Fórum permite a criação de um grupo temático de discussão por tópicos a ser estabelecido pelo tutor do curso, que pode criar ou eliminar um fórum livremente. O estudante que se interessar pelo tópico que está sendo discutido pode integrar o fórum se assim o desejar e o tutor do curso permitir.

A ferramenta Bate-papo funciona de forma a oferecer o suporte a conversação síncrona entre estudantes e/ou tutores. Um ponto que vale ressaltar é que essas conversas podem ser registradas na base de dados residente do servidor e podem

ser consultadas por qualquer participante do curso devidamente registrado no TelEduc.

A ferramenta de Correio funciona somente dentro do ambiente do TelEduc, ou seja, o estudante após entrar no sistema tem acesso as mensagens que lhe foram enviadas. A partir desse momento, ele pode ler as mensagens e responder as que achar conveniente. É uma forma de comunicação assíncrona fornecida pelo ambiente.

A ferramenta Mural é bastante simples e consiste basicamente de uma página no ambiente do TelEduc onde qualquer participante do curso pode disponibilizar informações que achar interessante que outros possam ter acesso.

Por fim, do grupo de ferramentas de comunicação, temos a ferramenta Diário e a ferramenta Perfil. A primeira tem por finalidade servir como um espaço onde o estudante possa registrar suas impressões a respeito do curso ao qual está inserido, suas dificuldades e reflexões em geral a respeito de seu próprio processo de aprendizagem. É através dessa ferramenta que os tutores do curso podem também deixar registrado para o estudante comentários em geral. Já a ferramenta Perfil tem por objetivo ser um espaço onde o estudante possa tecer comentários a respeito de si mesmo e disponibilizar uma foto.

5.1.3 Ferramentas de administração

Passaremos agora a análise das ferramentas de administração disponibilizadas pelo TelEduc. Nesse grupo temos as ferramentas que auxiliam o tutor do curso a realizar o gerenciamento de inscrições de estudantes no curso, datas de início e término, entre outras funcionalidades de caráter mais burocrático do processo de formalização de um curso a distância. É importante citar que essas ferramentas são de acesso exclusivo aos tutores de um determinado curso, sendo o

acesso restrito aos estudantes do mesmo. É nesse grupo que temos a ferramenta Acessos, que permite ao tutor de um curso visualizar o número de acessos e o último acesso dos participantes do ambiente (figura 5.4), a freqüencia em um determinado período do curso estipulado pelo usuário (figura 5.5) e o acesso às diferentes ferramentas disponíveis. Temos que o objetivo maior de implementação dessa ferramenta, segundo Vieira (2002), é "distinguir o aluno calado e presente do aluno realmente ausente e essa diferenciação é extremamente importante no acompanhamento de um curso."

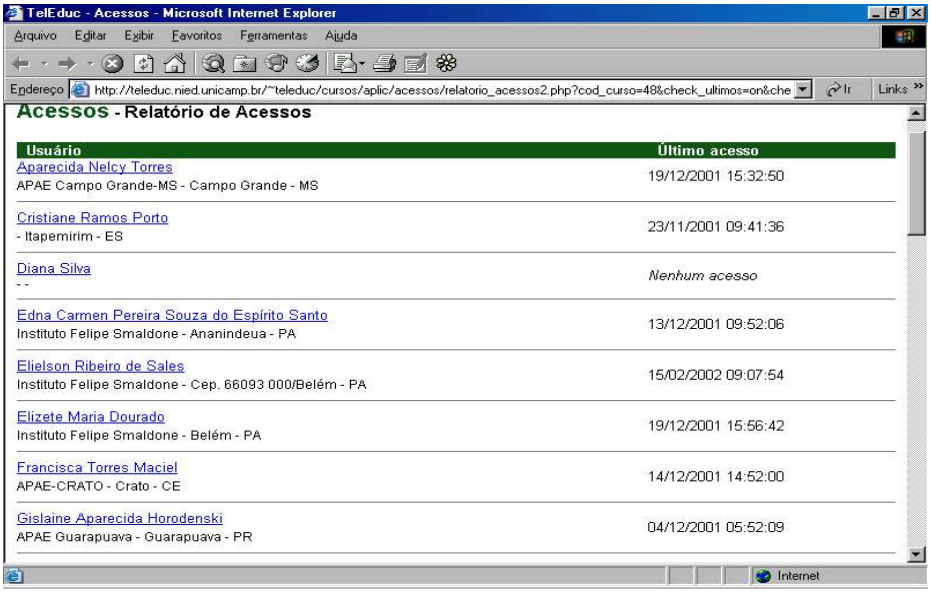


Figura 5.4 – Ferramenta Acessos – Último acesso dos participantes do ambiente – (Otsuka, Vieira, 2002).

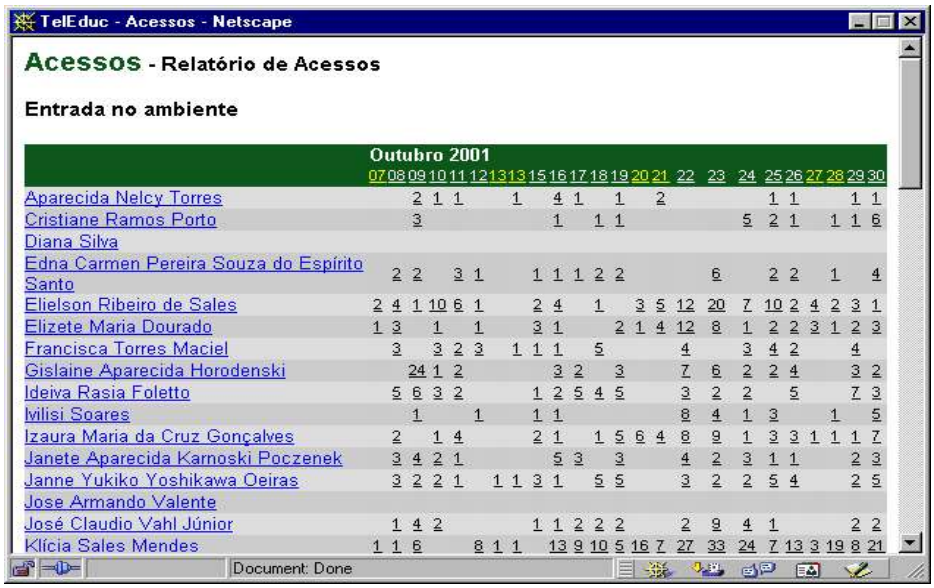


Figura 5.5 – Ferramenta Acessos – Freqüencia em um período do curso – (Otsuka, Vieira, 2002).

Por fim, podemos analisar a ferramenta InterMap (figura 5.6). Essa ferramenta funciona como um apoio aos participantes de um determinado curso a analisar as conexões desenvolvidas entre os vários estudantes que compõem o curso. Ela permite visualizar essas conexões estabelecidas de forma a auxiliar no processo de avaliação dos próprios estudantes. Ela tem, portanto, por objetivo fornecer subsídio "para auxiliar o participante a aprender a estrutura e o histórico da discussão, bem como as relações entre os participantes de um curso" (Vieira, 2002).

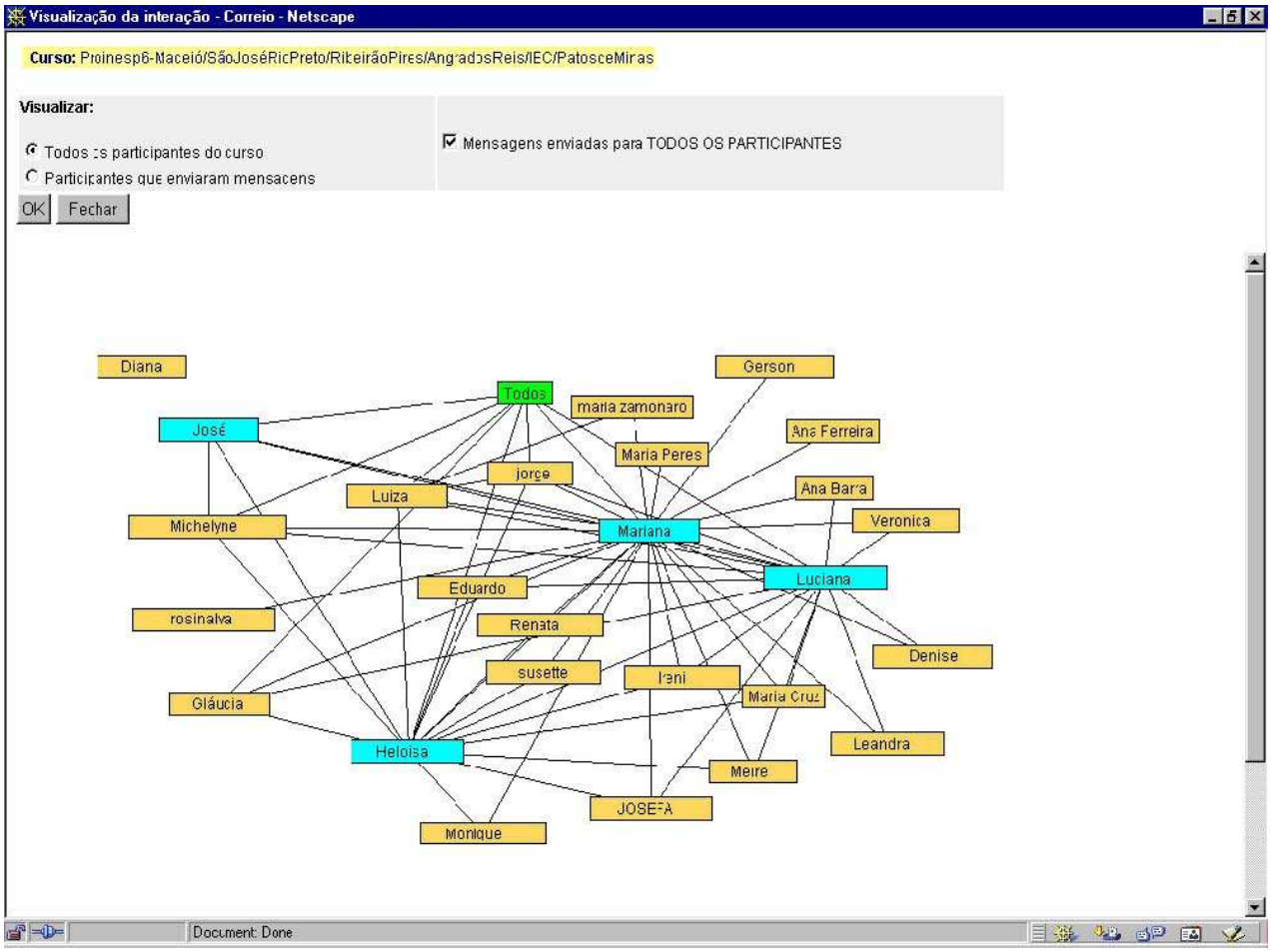


Figura 5.6 – Ferramenta InterMap – Visualização de interação – (Otsuka, Vieira, 2002).

Dessa forma, apresentamos aqui as principais características e funcionalidades relacionadas ao ambiente baseado na arquitetura cliente/servidor do TelEduc. É a partir da análise dessas ferramentas apresentadas que podemos descrever como o sistema SemiCode se relaciona com a arquitetura cliente/servidor e o TelEduc como caso de implementação dessa arquitetura.

5.2 Proposta híbrido TelEduc/SemiCode

O TelEduc é um ambiente focado no podemos chamar de ensino a distância (EaD) que foi concebido inicialmente para auxiliar no processo de formação de professores para a Informática na Educação (Vieira, 2002). Podemos

analisar sua arquitetura básica através da figura 5.1.

Após termos apresentando o TelEduc, podemos destacar alguns pontos nos quais a arquitetura P2P poderia auxiliar a arquitetura cliente/servidor do TelEduc na realização de algumas tarefas, de forma a facilitar e dar maior robustez às atividades desenvolvidas pelos usuários do sistema.

Temos que algumas atividades realizadas pelo TelEduc somente fazem sentido quando operadas a partir de uma estrutura hierárquica, como no caso das informações a respeito de matrículas num determinado curso ou no controle por parte do instrutor do acesso dos alunos e atividades desenvolvidas pelos mesmos. Essas atividades são hoje implementadas utilizando os recursos inerentes da arquitetura cliente/servidor e atendem, portanto, as necessidades de planejamento do sistema e da arquitetura computacional disponível para o mesmo.

No entanto, algumas das atividades que dizem respeito a interação entre usuários do sistema poderiam ser realizadas utilizando-se os recursos de redes P2P, de forma a aliviar o servidor de aplicações em relação a atividades que não necessitam serem mediadas pelo mesmo e oferecer maiores possibilidades de interação e conectividade entre os usuários. Estamos falando aqui mais precisamente das ferramentas de comunicação entre usuários no que diz respeito a troca de mensagens entre os mesmos. Sabendo-se que o TelEduc tem por um de seus objetivos registrar a dinâmica de interação entre seus usuários, temos que nesse formato, a rede P2P poderia enviar um registro das conversações estabelecidas entre pares para o servidor de aplicações, garantindo assim esse registro e mantendo as características iniciais do sistema.

Outra aplicação que poderia certamente se integrar ao TelEduc e auxiliar aos próprios usuários é a troca de arquivos a partir de diretórios compartilhados diretamente das máquinas dos usuários. Caberia, portanto, ao usuário submeter ou não um determinado arquivo ao servidor do TelEduc conforme necessidade de registro ou avaliação do conteúdo produzido. De qualquer forma, parte do material que não é necessário ao sistema, mas que pode ser de interesse do usuário disponibilizar, poderia ser feito através da arquitetura P2P.

Os grupos de pares utilizarão os cadastros de cursos do TelEduc para formarem seu grupo de interação, os registros das conversações serão enviados para os servidores do TelEduc e a troca de arquivos funciona de forma independente, permitindo interação entre os usuários de forma livre.

Dessa forma, propomos um sistema híbrido, que pode ser visto na figura 5.7, entre TelEduc e SemiCode, onde os recursos de conversação entre pares e troca de arquivos estivessem utilizando de redes P2P para a realização de suas tarefas, complementando assim os recursos tecnológicos disponíveis pelo TelEduc para a execução das atividades educacionais a que se propõem.

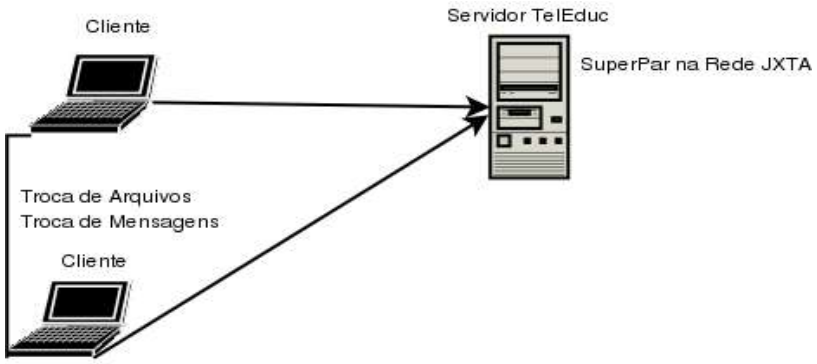


Figura 5.7 – Proposta TelEduc/SemiCode.

5.3 Resumo

Um fator de fundamental relevância que deve ser considerado aqui é o objetivo de uso proposto pelos dois sistemas, o que certamente nos levará a compreender a diferença conceitual entre ambos.

O ambiente TelEduc foi concebido para a formação de professores no âmbito da informática na educação e, portanto, foi um ambiente pensado de forma a propor um conteúdo específico a ser estudado por um grupo de estudantes com uma necessidade bastante próxima de aquisição de conhecimentos. Logo, essa característica levou o sistema a criar mecanismos que se desdobraram nas ferramentas apresentadas no item 5.1 que permitissem

a realização dessa formação de professores à distância, permitindo que os mesmos pudessem seguir os tópicos planejados pelos tutores de um curso e utilizassem as ferramentas de comunicação para conhecerem ou interagirem com seus pares no curso em busca de maiores infomações a respeito das tarefas que são propostas para resolução pelos estudantes. Logo, faz aqui todo sentido que as informações do ambiente sejam centralizadas na estrutura cliente/servidor, sendo que o servidor hospedará o ambiente de colaboração, receberá o conteúdo produzido pelos estudantes e dará suporte a interação dos usuários devidamente cadastrados no sistema.

O ambiente SemiCode foi concebido para facilitar a interação e a circulação da informação entre pares de um determinado grupo que tem objetivos afins, seja na realização de um projeto em âmbito empresarial, seja na pesquisa de um determinado assunto em âmbito acadêmico. O sistema permite, portanto, criar uma metáfora de um ponto de encontro virtual, os pares em comum envolvidos nesses projetos possam encontrar os outros pares no momento em que esses estão diretamente trabalhando em seus ambientes, possam acompanhar a evolução de material produzido pelos pares diretamente em suas máquinas de trabalho, avaliar e serem avaliados por outros pares presentes no ambiente no que condiz a sua própria reputação.

Logo, como poderíamos imaginar, temos que numa arquitetura cliente/servidor a relação entre os usuários do sistema e o sistema se dá de uma forma mais hierárquica e centralizada, e na arquitetura *peer-to-peer* uma relação mais emergente, partindo dos pares do sistema. Temos, portanto, dois ambientes que se complementam como ferramentas de interação e de trabalhos colaborativos em rede.

É, portanto, a partir deste fato que propomos um sistema híbrido, que possa utilizar alguns dos recursos das redes P2P de forma a ampliar as possibilidades de conectividade entre usuários e aliviar a carga dos servidores de aplicação do sistema.

Capítulo 6 – Conclusão

Após termos desenvolvido o presente estudo e termos implementado um modelo de aplicação de suporte à colaboração em redes P2P, podemos efetivamente analisar os pontos que nos levaram a esse trabalho e a forma como os mesmos foram tratados ao longo do tempo.

Dois pontos são fundamentais na concepção teórica e na motivação que gerou o interesse no presente estudo. Um deles é a possibilidade de que as redes P2P possam romper com o "paradigma do curso descente" (Oram, 2001, pág.10) em que o acesso atual a sites provedores de conteúdo da Internet se enquadra e o segundo é trabalhar com uma ferramenta que possa impulsionar a produção de conteúdo e a disponibilização do mesmo nas margens das grandes redes, privilegiando dessa forma o usuário dessas margens e não apenas a estrutura técnica que detém os recursos capazes de fornecerem grandes servidores com alta performance.

Olhando para o sistema SemiCode como o mesmo foi implementando, ou seja, utilizando o modelo de protocolos propostos pelo projeto JXTA e a linguagem Java, reconhecidamente multiplataforma, podemos notar que o mesmo se propõe a oferecer uma alternativa técnica a essa ruptura do curso descente de acesso ao conteúdo. Sendo um aplicativo que busca uma interface onde procuramos privilegiar o acesso a um outro par e não apenas às funcionalidades de busca de arquivos ou troca de conteúdo, temos que o aplicativo construiu seu foco dentro do âmbito de um pequeno grupo de usuários envolvidos num mesmo projeto, ou com foco empresarial ou educacional, onde a colaboração informal torna-se uma ferramenta de trabalho capaz de auxiliar no desenvolvimento de atividades relacionadas aos projetos em comum dos usuários.

Dessa forma o estudo realizado desdobra-se num sistema que tem sua atuação nas bordas da rede de informação, onde se tem a necessidade de criar condições que promovam a circulação da informação entre pares que não tem a necessidade de estarem conectados num mesmo servidor para se encontrarem e colaborarem entre si.

Para a construção desse sistema, o SemiCode, procuramos estudar desde os vários tipos de protocolos relacionados a redes P2P disponíveis até várias outras implementações de sistemas que usam o P2P como forma de trocarem informações entre pares. Essa pesquisa deu a base de sustentação de uma construção mais sólida em relação às alternativas já existentes. Pudemos analisar módulos de interação que possuem bom aproveitamento por parte dos pares da rede em outras implementações e averiguar até que ponto esses mesmos módulos poderiam se adaptar a proposta do SemiCode. Dessa análise derivaram os módulos de Chat, Questões e Troca de Arquivos.

Um tema bastante recorrente em sistemas de colaboração nos últimos anos tem sido a reputação, ou um sistema de confiança entre usuários, que permita dar maior crédito aos usuários que colaboram mais com a rede ou que cumprem acordos estabelecidos por meio de mensagens eletrônicas, como o caso do eBay, visto no capítulo 3. Há diversas propostas de sistemas de reputação atualmente em uso, sendo que algumas tendem mais para a informalidade da avaliação dos pares e outras para métricas mais objetivas com base em dados estatísticos abstraídos do próprio uso das redes de colaboração. Procuramos analisar algumas alternativas e propor um sistema de simples compreensão e uso por parte dos usuários do sistema de forma a valorizar a avaliação de terceiros a respeito do trabalho de um par. Dessa forma, estamos criando um mecanismo de avaliação da rede de colaboração para um trabalho desempenhado por um de seus pares. Nesse ponto pudemos dar um destaque para informações do perfil dos pares da rede, de forma a que os mesmos pudessem ter um espaço onde inserir algumas informações pessoais mas que poderiam auxiliar na colaboração entre pares e no encontro que os mesmos pudessem realizar através da rede. É portanto este um formato onde um par na rede recebe um *feedback* de sua rede de colaboração, informação relevante que pode efetivamente posicioná-lo em relação a forma como seu trabalho tem sido visto.

Além dessa proposta de implementação, procuramos analisar como esse sistema se relaciona com um sistema de colaboração em rede baseado na arquitetura cliente/servidor, o TelEduc. Essa análise comparativa tornou-se bastante rica no sentido de evidenciar que a soma dos modelos de

arquitetura em rede como proposta para um sistema de colaboração híbrido tende a privilegiar a interatividade entre usuários e uma maior independência e conectividade entre os mesmos, sem perder métricas necessárias para a coordenação de cursos disponibilizados de forma digital e ambientes de colaboração que propõem tarefas específicas a serem desempenhadas pelos usuários do sistema.

6.1 Sugestões de trabalhos futuros

Gostaríamos de deixar registrado neste trabalho algumas sugestões de trabalhos futuros que pudessem dar continuidade na proposta de ambiente de suporte à colaboração em redes P2P oferecida pelo SemiCode.

Inicialmente, propomos a criação de um mecanismo que permita fornecer aos usuários do sistema a opção de salvar sessões de bate-papo entre pares num modelo que fosse disponibilizado para a rede de colaboração como um todo. Basicamente, estamos falando aqui de um mecanismo que permita usarmos metadados para a classificação e identificação das sessões de bate-papo para fácil recuperação e aproveitamento das informações por parte dos pares na rede. O mesmo se aplica para sessões de questões entre pares, ou seja, permitir que os mesmos pudessem salvar essas informações criando dessa forma espécies de FAQs (*Frequently Answered Questions*) pessoais.

Outra proposta seria a criação de um módulo que permita a um determinado par descrever as atividades em que está envolvido no momento, ou seja, o trabalho que está envolvido nessa sessão de uso, seria como uma espécie de um mural de trabalhos sendo desenvolvidos na rede de colaboração de pares. Esse mural ficaria disponível juntamente ao perfil do par nos módulos de interação do SemiCode. Dessa forma, quando um determinado usuário desejasse ter mais informações de um respectivo par, ele também teria acesso a informações atualizadas a respeito dos trabalhos em que esse par está envolvido.

6.2 Perspectivas

O objetivo maior deste trabalho era estudar e oferecer uma proposta de sistema de colaboração em redes P2P que pudesse carregar conceitos que

auxiliassem na independência dos pares e na circulação da informação entre os mesmos. Realizada a presente implementação e apresentação do trabalho, temos agora como horizonte a implantação do SemiCode em comunidades de colaboração, o mapeamento e a identificação de mudanças a serem feitas no sistema, bem como a análise do comportamento das redes. Fica, portanto, uma nova fase onde teremos a interação do SemiCode com sua comunidade de usuários.

Apêndice A – Códigos

A.1 – Implementação Exemplo JAL

```
1           import net.jxta.endpoint.Message;
2           import net.pkg.jal.*;
3           import java.io.*;
4           public class TestApp {
5           private static Peer me;
6           public static void main (String args[]) {
7           me = new EZMinimalPeer();
8           try {
9             me.boot("MDE" + args[0]);
10            me.displayPeers();
11            me.displayGroups();
12            me.createGroup("MDE");
13            me.displayGroups();
14            me.joinGroup("MDE");
15            me.displayPeers();
16            me.displayGroups();
17            //finally try out getJoinedGroups
18            String a[] = me.getJoinedGroups();
19            System.out.println("getJoinedGroups:");
20            for(int i = 0; i < a.length; i++){
21                System.out.println(a[i]);
22            }
23            System.exit(0);
24          }catch (Exception e) {
25            e.printStackTrace();
26            System.exit(0);
27          }
28          System.exit(0);
29          }
30          }
```

Código 1 – Exemplo de implementação utilizando o JAL (fonte: http://ezel.jxta.org/jal.html).

O código 1 exemplifica o uso do JAL. Podemos notar que ele utiliza a classe EZMinimalPeer para acessar os métodos que permitem a realização das atividades básicas das primitivas P2P. Inicialmente, o programa inicializa o objeto me na linha 7 que utiliza a interface Peer, como declarado na linha 5. Através desse objeto, o programa inicializa o par atual na rede P2P informando que o mesmo fará parte de grupo de pares chamado MDE, linha 9; exibe os pares conectados no presente momento na linha 10; exibe os grupos de pares que podem ser identificados na linha 11; cria o grupo de pares MDE na linha 12; exibe novamente uma listagem dos grupos existentes para podermos perceber a presença do novo grupo

criado na linha 13; junta-se ao grupo MDE na linha 14; e na linhas 15 e 16 exibe novamente os pares conectados e os grupos de pares presentes na rede; da linha 18 a 22 o programa busca detectar quais são os grupos de pares ao qual o par atual está conectado.

A.2 – Estrutura de Arquivos SemiCode

O SemiCode trabalha com estruturas de arquivos que permitam utilizarmos serialização de objetos. Basicamente, utilizamos uma classe que implementa a serialização e que contém todos os campos de registro que pretendemos armazenar num determinado arquivo. A partir disto, utilizamos as classes **ObjectOutputStream** que converte qualquer objeto que seja **Serializable** (que é o caso de uma instância da classe que contém todos os campos de registro) em um fluxo de bytes que pode ser armazenado em um arquivo e **ObjectInputStream** que permite reconstruir o objeto original a partir do arquivo e, dessa forma, recuperar as informações armazenadas.

A seguir, podemos analisar como esse processo é feito no SemiCode na estrutura de arquivo que permite gravarmos as informações relacionadas ao perfil de um usuário. Inicialmente, temos uma classe que contém os registros que iremos armazenar em arquivo. Como podemos ver no código 2 na linha 2, essa classe implementa a serialização, o que permite transformar as variáveis que a mesma contém e que são as informações de perfil de um usuário em um fluxo de bytes que vai ser armazenado em arquivo e recuperado posteriormente.

```
1       import java.io.Serializable;
2       public class RegistroPerfil implements Serializable {
```

Código 2 – Implementação de classe serializável de registro de perfil.

Dessa forma, podemos ler os dados que foram armazenados em arquivo através da estrutura exibida no código 3.

```
1       RegistroPerfil registroLocal;
2       in = new ObjectInputStream(new FileInputStream(new
                        File(diretorio,"perfil.dat")));
```

```
3       registroLocal = (RegistroPerfil) in.readObject();
```

Código 3 – Recuperando um objeto serializado.

A linha 1 do código 3 instancia um objeto RegistroPerfil e o mesmo é inicializado com o objeto lido do arquivo perfil.dat (linha 2) e tipado com a classe RegistroPerfil. De tal maneira, temos as informações do perfil do usuário recuperadas e prontas para serem exibidas para o usuário local do sistema ou serem formatas em mensagens a serem enviadas pela rede P2P para pares remotos.

O mecanismo de gravação das informações em arquivos é bastante simples e pode ser visualizado no código 4, que consiste basicamente de enviarmos o objeto serializável para a saída de arquivos. Na linha 1, temos a inicialização do objeto de saída dos dados, que permite redirecionar o fluxo para um arquivo. Na linha 2, temos a inicialização do objeto serializável com as informações que desejamos armazenar e nas linhas 4 e 5 a gravação efetiva em disco.

Vale frisar que é essa mesma estrutura que é utilizada para gravar as informações de reputação de um determinado usuário do sistema.

```
1       output = new ObjectOutputStream(new FileOutputStream
   (diretorio+"/perfil.dat"));
2       registroLocal = new RegistroPerfil(valorCampos[0],
   valorCampos[1],valorCampos[2],valorCampos[3],valorCampos
[4],valorCampos[5],valorCampos[6],valorCampos[7]);
3       try{
4             output.writeObject(registroLocal);
5             output.flush();
6       }
7       catch (IOException io){
8             closeFile();
9       }
```

Código 4 – Gerando o fluxo de bytes do objeto serializável e redirecionando para o arquivo.

A.3 – Estrutura de Dinamica SemiCode

```
1       public static ArrayList pilhaMensagensChat = new
  ArrayList(1);
2       public static ArrayList pilhaSendersChat = new ArrayList
  (1);
3       if (type.compareTo("chatMessage")==0){
4             String peer,conteudo;
5             peer=mensagem.getString("peer");
6             conteudo=mensagem.getString("content");
7             pilhaMensagensChat.add(peer);
8             pilhaMensagensChat.add(conteudo);
9             if (pilhaSendersChat.size()==0){
10               pilhaSendersChat.add(peer);
11               chat = new ChatGUI
                    (grupo,peer,pilhaSendersChat,pilhaMensagensChat,
                    eu);
                    }
12            else{
13               int flag=0;
14               for(int i=0;i<pilhaSendersChat.size();i++){
15                   String verificaPeer = new String
                      (String.valueOf(pilhaSendersChat.get(i)));
16            if (verificaPeer.compareTo(peer)==0)
17                  flag=1;
18            }
19      if (flag==0){
20          pilhaSendersChat.add(peer);
21          chat = new ChatGUI
                (grupo,peer,pilhaSendersChat,pilhaMensagensChat,eu);
22          }
23        }
24      }
```

Código 5 – Pilha de mensagens de chat recebidas pela rede P2P.

Temos no código 5 que as linhas 1 e 2 inicializam as estruturas dinâmicas ArrayList. Na linha 3, o programa verifica se o tipo de mensagem que chegou contém em seu cabeçalho uma informação que a referencia com a

interação do tipo chat. Das linhas 4 a 6, retiramos as informações importantes a respeito do conteúdo da mensagem e do par que a enviou e as armazenamos em variáveis locais. As linhas 7 e 8 são as responsáveis por armazenarmos as informações relevantes da mensagem na estrutura ArrayList, como essa estrutura tem alocação dinâmica, isso significa que não precisamos declarar um número fixo de mensagens que podemos armazenar. Da linhas 9 a 11, realizamos um teste para verificar se temos algum par em conversação com o usuário no momento atual, se não estiver, o que é denotado pela pilha estar vazia, o par é somado na pilha de conversações em aberto e uma janela de chat é aberta para o usuário do SemiCode. Das linhas 12 a 23, verificamos se a mensagem que acabou de chegar é de um par que já se encontra em conversação com o usuário do sistema ou de um novo par que deseja estabelecer conversação, se for de um novo par, uma nova janela é aberta, permitindo assim um novo fluxo de mensagens entre dois pares na rede. Se não for um novo par, a classe que apresenta a janela aberta de conversação entre os dois pares irá se responsabilizar por ler constantemente as estruturas de dados em busca de mensagens relativas a conversação em aberto e atualizar a interface com o usuário.

A.4 – Monitoramento da Rede

Quando apresentamos uma interface gráfica ao usuário em que oferecemos opções de interação, como campos de texto ou botões, a classe que é responsável por exibir a janela implementa a interface ActionListener que permite implementar um método chamado actionPerformed, responsável por monitorar qualquer ação do usuário sob a interface gráfica. Esse método fica constantemente monitorando as ações sob a tela e se tentarmos realizar alguma outra atividade sem a implementação de multithread, a interface gráfica irá congelar enquanto essa outra atividade é executada. Essa característica certamente inviabilizaria os módulos de interação propostos para o SemiCode.

A implementação das atividades em paralelo do SemiCode foi baseada num tutorial sobre programação de threads e o uso do Swing (http://java.sun.com/products/jfc/tsc/articles/threads/ threads1.html).

Para isso usamos um método da classe SwingUtilities chamado invokeLater(), que permite que um processo seja executado na thread principal, a mesma que está monitorando os objetos da interface gráfica com o usuário e retorne imediatamente, sem esperar que o processo tenha terminado sua execução. Essa característica permite que a tela não fique congelada enquanto o processo está sendo executado e permita atualizar a mesma sem um efeito desagradável para o usuário. Como o monitoramento da rede é um processo que é executado continuamente e com algum intervalo de tempo, usamos aqui também a classe Timer, que cria uma thread que executa algum código uma ou mais vezes em um determinado espaço de tempo. No código 6, podemos visualizar o código do SemiCode que implementa essa questão.

```
1       javax.swing.SwingUtilities.invokeLater(new Runnable() {
2           public void run() {
3             show();
4             int delay=4000;
5             ActionListener taskPerformer = new ActionListener() {
6                   public void actionPerformed(ActionEvent evt) {
7                         exibeMensagens();
8          }
9        };
10            new Timer(delay,taskPerformer).start();
11            }
12      });
```

Código 6 – Estrutura de monitoramento de mensagens multithread.

Analisando o Código 6, podemos perceber que a exibição da interface gráfica ao usuário, que é feita na linha 3, está dentro do método invokeLater, ou seja, é através dessa declaração que temos a possibilidade de iniciar uma outra atividade e retornar imediatamente para o monitoramento das ações do usuário sob a interface enquanto essa outra atividade é processada. No caso, essa outra atividade é caracterizada aqui pela chamada do método exibeMensagens() que é o responsável por retirar das estruturas dinâmicas de dados relatadas no item A3 as informações que devem ser exibidas ao usuário dentro da atividade de interação atual na qual ele se encontra, seja chat, troca de arquivos, etc.. Essa chamada de método é

realizada dentro do método actionPerformed, que está diretamente ligado a classe Timer e, portanto, a cada intervalo de tempo determinado pela linha 4 o método actionPerformed é chamado, logo o método exibeMensagens é executado nesse mesmo intervalo de tempo, o que permite atualizar a interface gráfica com o usuário com as novas mensagens de chat que chegam e são armazenadas nas ArrayLists do item A3.

Bibliografia

BARABASI, Albert László, 2002. *Linked: the new science of networks.*

DAMIANI, Ernesto, VIMERCATI, Sabrina De Capitani di, PARABOSCHI, Stefano, SAMARATI, Pierangela, VIOLANTE, Fabio 2002. *A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks.* CCS'02, November 18-22, 2002, Washington, DC, USA. ACM 1-58113-612-9/02/0011.

EIKEMEIER, Claus, LECHNER, Ulrike, BEYERLEIN, Florian, LODDE, Michael, 2003. *The iKnow Tool: connecting software engineers with peer-to-peer software ad-hoc collaboration.* At 2nd Workshop on Computer Supported Software Engineering 2003 (CSSE'03), Benevento, Italy. Published in: *Cooperative Methods and Tools for Distributed Software Processes,* pp. 120-134, Publisher: Franco Angeli, Milano, Italy, ISBN 88-464-4774-3.

HALM, Mike, VALENTINE, Alex, 2002. *LionShare: an academic P2P community.* Disponível em:http://lionshare.its.psu.edu

IONESCU, Bogdan, BINDER, James, IONESCU Dan, 1999. *A distributed architecture for collaborative applications.* Communications, Computers and Signal Processing, 1999 IEEE.

JOHNSON, Steven, 2001. *Cultura da Interface. Como o computador transforma nossa maneira de criar e comunicar*. Jorge Zahar Editor.

JOHNSON, Steven, 2003. *Emergência. A dinâmica de rede em formigas, cérebros e softwares.* Jorge Zahar Editor.

KAMVAR, Sepandar D., SCHLOSSER, Mario T., GARCIA-MOLINA, Hector, 2003. *The Eigentrust Algorithm for Reputation Management in P2P Networks.* WWW2003, May, 20-24, 2003, Budapest, Hungary. ACM 1-58113-680-3/03/0005.

LÉVY, Pierre, 1993. *As tecnologias da inteligência. O futuro do pensamento na era da informática.* Editora 34.

NEJDL, Wolfgang et al., 2002. *Edutella: a P2P networking infrastructure based on RDF.* WWW2002, May, 7-11, 2002, Honolulu, Hawaii, USA. ACM 1-58113-449-5/02/0005.

ORAM, Andy (organizador), 2001. *Peer-to-peer: o poder transformador das redes ponto-a-ponto.* Editora Berkeley.

RATNASAMY, Sylvia, FRANCIS, Paul, HANDLEY, Mark, KARP, Richard, SHENKER, Scott, 2001. *A scalable content-addressable network.* SIGCOMM'01.

ROHRS, Christopher, 2001. *Limewire design.* Fonte: http://www.limewire.org/project/www/design.html.

ROSENBERG, Doug, SCOTT, Kendall, 2000. *Driving Design with Use Cases.* Fonte: www.sdmagazine.com.

ROWSTRON, Antony, DRUSCHEL, Peter, 2001. *Pastry: Scalable descentralized object location and routing for large-scale peer-to-peer systems.* Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware, 2001).

SCHODER, Detlef, FISCHBACH, Kai, 2003. *Peer-to-peer prospects.* Communications of the ACM, February 2003/Vol.46 No.2.

STOICA, Ion, MORRIS, Robert, KARGER, David, KAASHOEK, M. Frans, BALAKRISHNAN, Hari, 2002. *Chord: a scalable peer-to-peer lookup service for Internet applications*. SIGCOMM'01.

TRAVERSAT, Bernard, ARORA, Ahkil, ABDELAZIZ, Mohamed, DUIGOU, Mike, HAYWOOD, Carl, HUGLY, Jean-Christophe, POUYOUL, Eric, YEAGER, Bill, 2003. *Project JXTA 2.0 Super-Peer Virtual Network.* Fonte: www.jxta.org.

VIEIRA DA ROCHA, Heloísa, 2002. *Projeto TelEduc: Pesquisa e Desenvolvimento de Tecnologia para Educação à Distância.* Fonte: http://www.nied.unicamp.br

ZHAO, Ben Y., KUBIATOWICZ, John, JOSEPH, Anthony D., 2001. *Tapestry: An infrastructure for fault-tolerant wide-area location and routing.* Report No. UCB/CSD-01-1141. University of California, Berkeley.