Usando o modelo 3C de colaboração e Vygotsky no ensino de programação distribuída em pares.

Robson M. Borges[1], Sérgio Crespo C. S. Pinto[1], Jorge L. V. Barbosa[1], Débora N. F. Barbosa[2]

[1]Programa Interdisciplinar em Computação Aplicada - PIPCA – UNISINOS
Caixa Postal 15.064 – 91.501-970. São Leopoldo – RS – Brasil

[2]Curso de Ciência da Computação - Centro Universitário La Salle – UNILASALLE
Rua Victor Barreto, 2288 – 92.010-000. Canoas-RS-Brasil

byrobson@yahoo.com.br, {crespo, jbarbosa}@unisinos.br,
nice@unilasalle.edu.br

Abstract

Abstract. *The software engineering area is relatively recent and we do not have an ideal structured form for the software development. One of the software study lines give importance the agile methodologies as for instance the Extreme Programming that it distinguishes itself by reaching greater success. One of the most important practical features of the "XP" related to reaching quality and productivity is the pair programming. Alongside, we geographically see a spread in the formation of dispersed teams groups. This work present a software platform that makes possible the distributed programming in pairs called RemoteAPP. Related works is presented too.*

Resumo. *A engenharia de software é uma área de conhecimento relativamente recente. Ainda não possuímos uma forma idealmente definida para o desenvolvimento de software. Uma das linhas de estudo valoriza as metodologias ágeis. Dentre as quais, a extreme programming, também conhecida como XP, alcança maior sucesso. Uma das práticas mais importantes da XP no sentido de alcançar qualidade e produtividade é a programação em pares. Em paralelo, vemos uma expansão na formação de grupos de trabalho dispersos geograficamente. Este trabalho apresenta uma plataforma computacional que viabiliza a programação em pares distribuída denominada RemotePP. Aspectos de modelagem da ferramenta são apresentados, além de uma comparação com outros trabalhos relacionados.*

1. Introdução

O aumento dos ambientes colaborativos em empresas produtoras de software desencadeia um novo perfil do aluno que entra nos cursos de computação: "o perfil de trabalhar em grupo". Para isto, é necessário aliar o uso de ferramentas colaborativas e teorias pedagógicas que reforcem a colaboração entre pares e acentuem a colaboração em atividade de desenvolvimento de software. Por outro lado, desenvolver software é uma atividade altamente intelectual e que exige troca de conhecimento constante entre as pessoas envolvidas. Tal necessidade por interação é evidenciada por algumas práticas de desenvolvimento, como a "programação em pares". É necessário, ainda, que os pares sejam permutados a cada nova atividade, de forma a disseminar o conhecimento adquirido entre os aprendizes.

Para Williams (2000), "o olho humano tem uma grande capacidade de não enxergar aquilo que não quer ver". Da mesma forma, um aluno iniciante pode ignorar completamente um erro gritante que seria rapidamente percebido por um observador. Pesquisas realizadas pelos autores revelam que dois programadores juntos chegam a uma solução mais rápida e com maior qualidade do que dois programadores trabalhando separadamente na resolução de problemas.

Hanks (2005) identifica que as iniciativas para viabilizar programação por pares distribuída utilizam aplicações não orientadas para este fim. São utilizados: compartilhamento de *desktop,* e-mail, troca instantânea de mensagens, voz sobre IP, repositórios de arquivos compartilhados e videoconferência. Tais aplicações possuem objetivos específicos e não são orientadas à programação. Neste contexto, torna-se importante à existência de um ambiente integrado que viabilize diferentes formas de comunicação orientadas ao ensino de programação, por pares, de forma distribuída e colaborativa. Neste sentido, este trabalho apresenta a ferramenta RemotePP. Esta consiste em um ambiente colaborativo para auxílio à programação distribuída em pares. A sua principal contribuição está na possibilidade de utilização de ferramentas interativas que tornam mais eficiente o processo de colaboração. Além disso, do ponto de vista pedagógico, trabalhar em pares, valendo-se de um instrumento mediador, segundo o princípio sócio-cultural da teoria vygotskyana, influi nas interações, pois, em geral, o indivíduo costuma explicitar suas estratégias durante a resolução de tarefas compartilhadas.

Este artigo está organizado em 5 seções. A seção 2 apresenta conceitos importantes para compreensão da proposta. A seção 3 apresenta os trabalhos relacionados. Aspectos de modelagem da ferramenta RemotePP, bem como sua diferenciação quanto a outras propostas são apresentados na seção 4. As conclusões e trabalhos futuros são apresentados na seção 5.

2. Groupware / Modelo 3C e sua fundamentação pedagógica

Segundo Ellis (1991), CSCW ou *Groupware* é definido como o estudo dos sistemas que integram o processamento de informações com atividades de comunicação, de forma a identificar como os grupos trabalham e como a tecnologia pode ajudá-los a resolver tarefas em comum, reforçando a importância da comunicação, colaboração e coordenação [Ellis 1991].

Segundo Fuks (2003), a comunicação oportuniza a negociação e o estabelecimento de compromissos entre os participantes do grupo. Para utilizar o computador como ferramenta de

comunicação, é necessário que o sistema ofereça suporte à interação entre as pessoas. Deve haver controle entre os estados, eventos e diálogos de cada participante, conforme apresentado na Figura 1.

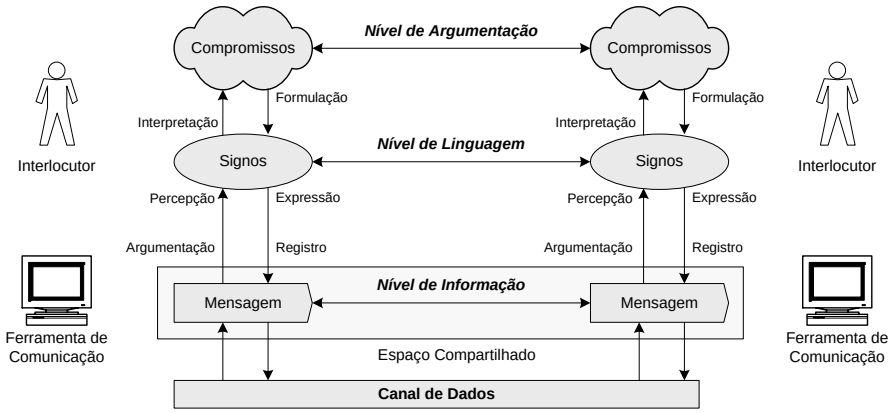

Figura 1 - Comunicação mediada por computador.

De acordo com o modelo, o emissor define a sua intenção (no nível mais abstrato de idéias) e a formula através de símbolos. Utiliza o computador para expressá-la numa mensagem e transmiti-la via canal de dados. Chegando ao destino, a mensagem é percebida e interpretada, de forma a modificar seus argumentos. Fuks estabelece dois tipos de classificação para as ferramentas de comunicação, podendo ser analisado quanto ao tipo de resposta e quanto ao nível de aplicação [Fuks 2003].

(a) Quanto ao tempo de resposta: podem ser síncronas e assíncronas.

(b) Quanto ao nível de aplicação:

- Sistemas de mensagens – suportam a troca de mensagens de texto entre usuários.
- Editores multi-usuários – onde códigos podem ser alterados por vários usuários.
- Decisão em grupo – oferecem mecanismos para tomada de decisão em grupo.
- Conferências – São em geral módulos com apoio de áudio e vídeo.
- Agentes Inteligentes – Sistemas de software autônomos.
- Sistemas de coordenação – permitem o controle e gerência das atividades do grupo.

Os autores estabelecem, ainda, um conjunto de funcionalidades importantes a sistemas de colaboração em tempo real (síncronos), conforme apresenta a Tabela 1:

Tabela 1. Funcionalidades de sistemas de colaboração síncronas

Funcionalidade	Definição
Ações	O conjunto de objetos e ações, realizadas e percebidas por todos.
Janelas	Áreas que estão compartilhadas e sincronizadas por todos os participantes on-line.
Cursos	Movimento do cursor é percebido e disponibilizado a todos.
Visão	Uma representação visual ou multimídia de parte do conteúdo compartilhado.
Interação	Ações percebidas de forma síncrona e assíncrona.

Funcionalidade	Definição
Seção	Período de interação síncrona.
Papéis	Conjunto de privilégios, atribuições ou responsabilidades de participante do grupo.

2.1. Desenvolvimento de Software Distribuído

A tendência pelo desenvolvimento de software através de grupos geograficamente distribuídos é justificada, segundo Herbsleb, pela redução de custos e busca por mão-de-obra qualificada. Os autores citam fatores como necessidades competitivas e a globalização das operações corporativas como atenuantes dessas tendências. Evidências indicam que o desenvolvimento distribuído costuma consumir tempo maior, quando comparado ao realizado por equipes geograficamente centralizadas, identificando a comunicação e coordenação como principais causas para esse aumento de tempo [Herbsleb 2001].

Carmel (1997) observa que a riqueza na comunicação pode ser medida em relação número de canais sensoriais utilizados pelas pessoas. Apenas 20% da comunicação é verbal. Isso leva a crer que a maior parte da comunicação ocorre de maneira não verbal, ou seja, através de gestos, timbre de voz, expressões faciais ou representações gráficas.

2.2. Programação em Pares Distribuída

Vários estudos têm sido realizados com o objetivo de avaliar as diferenças apresentadas, quanto à produtividade e qualidade, da programação em pares distribuída em relação à realizada "lado a lado". Em Stotts (2003) é citado como vantagens:

- Redução da necessidade de viagens;
- Os pares são forçados a manter registros eletrônicos de seus trabalhos e idéias;
- Os membros tendem a diminuir as conversas *off-topic*.

Schummer percebeu que os programadores devem estar aptos a se comunicar, ambos devem acompanhar o mesmo trecho de código, mas apenas um deve poder alterá-lo. A comunicação pode ocorrer por voz. Para os autores e co-autores, o contato visual não é necessário [Schummer 2001].

A maioria das formas utilizadas para co-autoria restringe ao compartilhamento de *desktop*. Desta maneira, ambos os programadores enxergam o mesmo ambiente de programação. Um assume o papel de *driver* e digitava o código, enquanto o outro apenas observava (*navigator*). A comunicação ocorre por voz, vídeo e *chat*. Quando é necessário mostrar alguma idéia na forma de diagramas, este é desenhado à mão em uma folha de papel e aproximado de uma câmera de vídeo. O uso da câmera de vídeo foi mais intenso no início dos estudos, com o passar do tempo, as duplas preferiram apenas utilizar comunicação por voz e *chat*.

2.3. Teorias pedagógicas que fundamentam o modelo *Groupware*

A interatividade está presente como elementos fundamentais no processo de construção do conhecimento do indivíduo. De acordo com Piaget o conhecimento não está no sujeito nem no objeto, mas ele se constrói na interação do sujeito com o objeto. É na medida em que o sujeito interage (e, portanto age sobre e sofre ação do objeto) que ele vai produzindo sua

capacidade de conhecer e vai produzindo também o próprio conhecimento [Franco 1996]. Já Vygotsky aborda este processo do ponto de vista de interatividade entre o indivíduo e o meio social. O princípio sócio-cultural da teoria vygotskyana nos traz um forte embasamento pedagógico para o trabalho em equipe, onde a importância da comunicação, colaboração e coordenação a fim de viabilizar o trabalho em grupo e obter melhores resultados, quando comparados ao trabalho realizado individualmente, é o elemento de sucesso.

Para Vygotsky, os processos mentais só podem ser entendidos se forem entendidos os instrumentos e os signos que medeiam esses processos. Além disso, o desenvolvimento cognitivo do ser humano não pode ser entendido fora do contexto social e cultural em que este se produz. E para entender o desenvolvimento cognitivo do ser humano, bem como os instrumentos e signos que medeiam esses processos em Vygotsky se apóiam no método genético experimental. Compreende-se então que, para Vygotsky, o modelo histórico-social é esboçado nas estruturas de mediação instrumental e social, que por sua vez internalizam, no ser humano, estruturas que possibilitam a interpretação do movimento quer pela passagem de ações realizadas no plano social, portanto inter-psicológicas, quer pela passagem de ações internalizadas (intra-psicológicas).

No caso de ferramentas de *Groupware* essas ações interpsicológicas e intra-psicológicas possuem uma dimensão não linear o que possibilita ao indivíduo uma postura exploratória maior. Desse modo as interações entre o grupo potencializam construções de conhecimento mais autônomos e criativos [Primo 1999]. Trabalhar em pares, valendo-se de um instrumento mediador, influi nas interações, pois, em geral, o indivíduo costuma explicitar suas estratégias durante a resolução de tarefas compartilhadas.

3. Trabalhos Relacionados

Esta seção avalia um conjunto de ferramentas semelhantes.

3.1. Sangam

O SANGAM foi desenvolvido por Ho (2004) na forma de um *plug-in* para o ambiente de desenvolvimento Eclipse da IBM. A ferramenta oferece sincronização de código-fonte entre mais de dois programadores. O SANGAM não bloqueia o texto durante uma seção. A arquitetura do SANGAM utiliza três componentes principais:

- Interceptador de eventos – Monitora o ambiente a fim de capturar todas as ações que o usuário (*driver*) realiza no editor de código Java da aplicação;
- Servidor de mensagens – É responsável pelo envio das mensagens ao *navigator*;
- Reprodutor de eventos – Realiza as ações recebidas pelo servidor de mensagem de modo a reproduzi-las no computador do *navigator*.

3.2. Flecse

O Flecse atua como um conjunto de ferramentas colaborativas e aplicáveis a todo o ciclo de vida de desenvolvimento de software [Dewan 1993]. A interface de todas as aplicações é baseada em linha de comando e console texto, conforme a seguir:

- O controle de versão;

- MShell: é um interpretador de comandos colaborativo;
- Teleconf: permite gravação, transmissão e execução de uma conversação por áudio;
- MDebug: possibilita a depuração de programas de forma compartilhada;
- CSI: é um mecanismo de inspeção assíncrono e síncrono;
- MEdit é um editor colaborativo que permite o bloqueio do código fonte.

4. RemotePP: um ambiente colaborativo de suporte a programação em pares distribuída

A ferramenta RemotePP tem como objetivo dar suporte à programação em pares e distribuída usando o modelo colaborativo 3C [Fuks 2003]. A primeira representação do sistema é apresentada na Figura 2. Na figura, um problema é apresentado a dois programadores localizados em regiões geográficas diferentes. Ambos devem encontrar uma alternativa de solução viável para o problema. Para tanto, recorrem a seus conhecimentos prévios ou experiências anteriores. Essas informações são trocadas entre os envolvidos de modo a encontrar uma solução única. Neste contexto, a RemotePP deve assumir papel de facilitador técnico capaz de permitir uma interação eficaz entre os programadores. A aplicação deve oferecer tanto suporte a tarefas em nível estratégico e cognitivo, como a comunicação, quanto em nível técnico, através de um ambiente de programação e diagramação de representações gráficas. O RemotePP tem como embasamento pedagógico o modelo sócio-cultural de Vygotsky, onde o processo de construção do conhecimento se dá através da interação do indivíduo com o meio, e as interações entre o grupo potencializam o desenvolvimento de um sujeito mais autônomo e criativo.

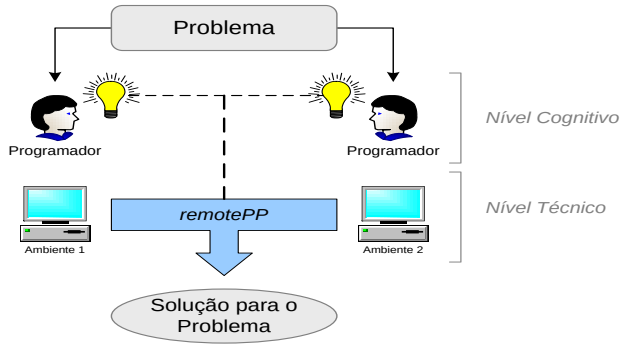

Figura 2 - Visão geral do sistema

4.1. Atores

Os atores na ferramenta podem assumir os seguintes papéis:

- Programador – Usuário geral do sistema. O termo "programador", neste contexto, também pode ser atribuído a "projetista". Não deve ser unicamente interpretado como codificador (pessoa responsável exclusivamente pela edição de código-fonte).
- Navigator e Driver – Papel assumido pelo programador quando se estabelece um protocolo entre as atividades desempenhadas pelo programador local e seu parceiro remoto. O Navigator acompanha atentamente o que o Driver está codificando ou

projetando. Pode oferecer alternativas de solução ou realizar pesquisas em bases de conhecimento como a Internet. Os papéis entre Driver e Navigator podem ser alternados durante a sessão. Esta convenção é utilizada para evitar conflitos ao editar documentos ao mesmo tempo.

- Remoto – Representa um ambiente do *Navigator*.

4.2. Arquitetura

A Figura Figura 3 apresenta organização dos módulos do RemotePP. Cada bloco representa a solução para um problema específico e a ligação entre os módulos indica o relacionamento entre os mesmos.

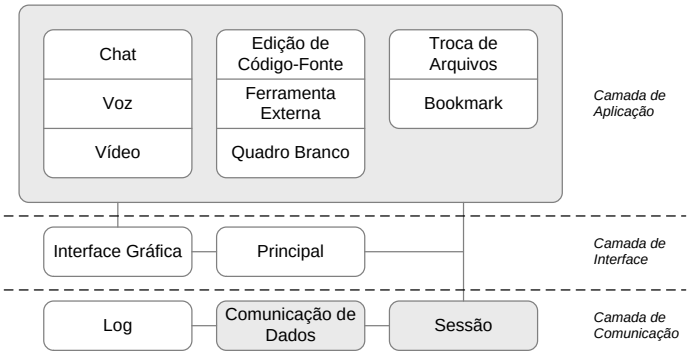

Figura 3 - Arquitetura do sistema

Comunicação de Dados é o módulo responsável pela troca de pacotes entre a aplicação local e o sistema que é executado remotamente. O módulo de ***Log*** é acoplado ao módulo de comunicação para oferecer uma forma única de persistência do tráfego gerado durante a comunicação armazenado todas as mensagens trocadas. O módulo **Sessão** representa o tempo em que dois computadores estão em comunicação síncrona. O módulo de **Interface Gráfica** consiste em *templates* e componentes comuns de interface gráfica com o usuário. O **Principal** é o módulo referente ao ponto de entrada da aplicação e oferece os mecanismos para o usuário realizar todas as atividades do sistema.

Os módulos abaixo compreendem as atividades de colaboração ente os usuários. Todos os módulos utilizam as definições da Interface Gráfica e dependem da existência das rotinas especificadas pelo módulo de Sessão. São eles:

- Chat, Voz e Vídeo – Compreende os módulos para troca de mensagens instantâneas de texto, comunicação por voz e comunicação através de videoconferência;

- Edição de Código-Fonte, Ferramenta Externa e Quadro Branco – Oferecem recursos para a edição de programas e o controle da execução de ferramentas externas, tais como compiladores por linha de comando. O módulo Quadro Branco disponibiliza uma forma de comunicação visual entre os programadores. Permite que possam ser expressos livremente pensamentos através de representações gráficas, sem que tenhamos o formalismo de uma linguagem específica;

- Troca de Arquivos e *Bookmark* – O módulo de troca de arquivos permite que arquivos binários alheios à aplicação sejam enviados ou recebidos. O fluxo e tamanho

dos pacotes são configurados de acordo com a largura da banda disponível. Já o módulo de *Bookmark* permite o compartilhamento de endereços de *links* da Internet utilizados frequentemente pelo usuário. Permite uma forma simples de conhecer os gostos pessoais do interlocutor.

4.3. Principais Interfaces do RemotePP

O módulo principal oferece o ambiente de trabalho do sistema na forma de um contêiner onde os componentes visuais da aplicação são dispostos. O protótipo da tela principal é apresentado na Figura 3 e sua interface foi projetada de forma a permitir que o usuário possua acesso direto a todas as principais funcionalidades de comunicação (à esquerda) e edição de código-fonte ou diagramas (à direita). À medida que as funcionalidades são ativadas, os botões são habilitados. Nesta figura, também se visualizar a interface de quadro branco compartilhado, usando vídeo conferência, onde os alunos podem interagir na confecção de diagramas.

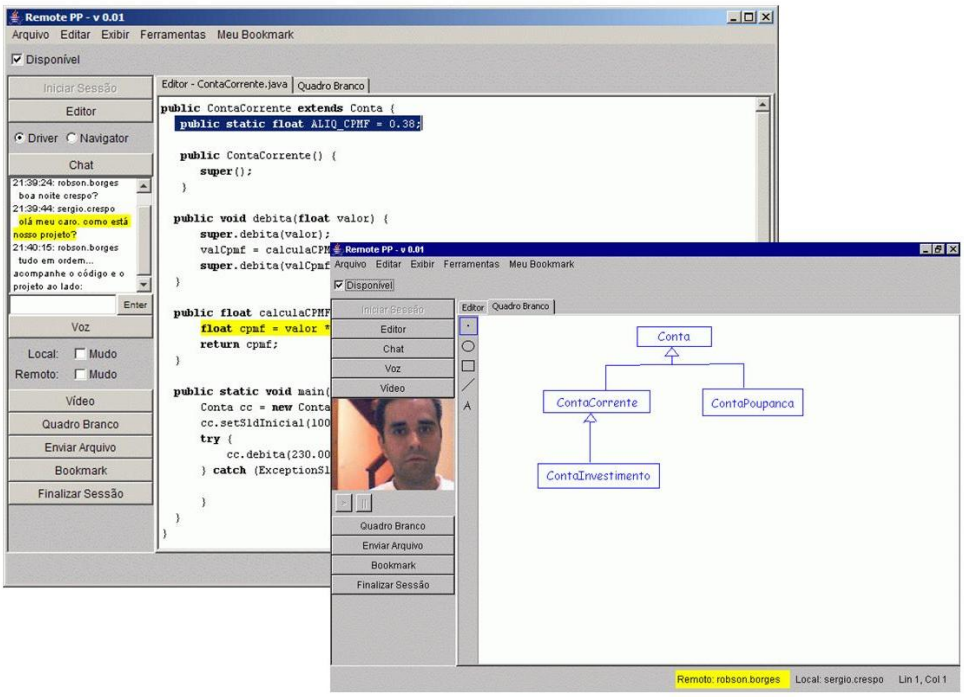

Figura 4 - Interface de Edição e Quadro Branco Compartilhado

O funcionamento do quadro branco é semelhante ao editor de código-fonte, com a diferença que, neste caso, ambos os programadores (local e remoto) podem alterar o artefato simultaneamente.

4.4. Diferencial do RemotePP

O diferencial do RemotePP frente a outros trabalhos relacionados na seção 3, concentra-se na preocupação com aspectos de comunicação, pois se percebe que este é o principal problema enfrentado por equipes geograficamente distribuídas. Isso ocorre, conforme, porque não é possível acompanhar as reações do interlocutor ao receber uma mensagem à distância. Numa conversa presencial tem-se maior probabilidade da compreensão da mensagem, pois é

possível avaliar o comportamento do interlocutor através de gestos, expressões faciais ou mudanças na tonalidade de voz [Kircher 2001].

O RemotePP apresenta, ainda, preocupação com recursos do ambiente de programação. Em especial, a necessidade de bloquear um dos usuários para edição do artefato. Porém, é importante que ambos possam acompanhar o *telecursor* do parceiro, a fim de compartilhar seu raciocínio e contribuir para a solução de um problema.

A Tabela 3 apresenta a justificativa das funcionalidades oferecidas pelo RemotePP. No quadro, é possível identificar os desafios propostos pelos autores, juntamente com o conjunto de ferramentas suportas no RemotePP.

Tabela 3. Justificativa das funcionalidades oferecidas pelo RemotePP

Perspectiva	Desafio	Funcionalidade
Comunicação [Ellis 1991]	Efetividade das interações a distância	Chat, Voz, Vídeo, Quadro Branco.
Relacionamento interpessoal [Stotts 2003]	Entender sobre: estilo, preferências pessoais e personalidade do parceiro.	*Bookmark* compartilhado.
Colaboração [Beck 2004]	Estabelecer um protocolo, no qual, enquanto uma pessoa codifica, o outro assume papel mais estratégico.	Bloqueio alternado do comando do editor de código fonte. Tele cursor. Rascunho compartilhado.
Infra-estrutura [Crowley 1990]	Permitir continuidade do trabalho, mesmo em caso de queda da rede.	Ambiente de programação mono e multi-usuários. Arquitetura não dependente de servidor centralizado.
Codificação [Dewan 1993]	Compartilhar a compilação, para obter auxílio para detecção de erros de lógica ou sintaxe.	Compilação com uso de *shell* compartilhado.

5. Conclusões e trabalhos futuros

O RemotePP visa auxiliar as atividades de programação em pares de forma distribuída. O ambiente entra em harmonia com o modelo 3C de cooperação onde a comunicação tem papel importante em processos colaborativos. O ambiente oferece um vasto conjunto de ferramentas que permitem uma maior interação e troca de documentação entre os pares. Toda a comunicação e atividades registradas em uma seção no RemotePP são armazenadas em formato XML de forma a futura análise por agentes inteligentes para a captura de perfis de desenvolvedores para a melhor formação de equipes, análise de estilos de programação, identificação de erros mais frequentemente cometido por uma das duplas.

O RemotePP permite que a compilação possa ser executado tanto no *driver* como no *navigator*, sendo que ambos podem visualizar a janela de erros e disparar a compilação em qualquer um dos terminais. Isto possibilita uma economia em termos de software, pois somente um dos participantes necessita ter o compilador.

Como trabalho futuro, espera-se desenvolver os agentes já citados e introduzir a figura do mediador, de forma a enriquecer ainda mais o modelo de colaboração entre os desenvolvedores.

6 Referências

Beck, K. (2004) Programação extrema explicada: acolha as mudanças. Bookman.

Carmel, E. (1997) "Thirteen assertions for globally dispersed software development research", Proceedings of the Thirtieth Hawaii International Conference on System Sciences, vol.3, no.pp.445-452 vol.3, 7-10.

Crowley, T., Milazzo, P., Baker, E., Forsdick, H., Tomlinson, R. (1990) "MMConf: an infrastructure for building shared multimedia applications". In Proceedings of the 1990 ACM conference on Computer-supported cooperative work, 329-342, Los Angeles, California, United States, ACM Press.

Dewan, P.; Riedl, J. (1993), Toward computer-supported concurrent software engineering. Computer, vol.26, no.1pp.17-27.

Ellis, C. A., Gibbs, S. J., and Rein, G. (1991). Groupware: some issues and experiences. ACM Communications Vol.34, No. 1 , 39-58.

Franco, Sérgio R. K. (1996) O construtivismo e a educação. Porto Alegre: Mediação.

Fuks, H., Raposo, A, Gerosa, M. (2003)"Do Modelo de Colaboração 3C à Engenharia de Groupware". Simpósio Brasileiro de Sistemas Multimídia e Web - WEBMIDIA.

Hanks, B. (2005). "Student performance in CS1 with distributed pair programming". In Proceedings of the 10th Annual SIGCSE Conference on innovation and Technology in Computer Science Education. ITiCSE '05. ACM Press, New York, NY, 316-320.

Herbsleb, J.D.; Moitra, D. (2001), Global software development, IEEE Software, vol.18, no.2pp.16-20, Mar/Apr 2001

Ho, C., Raha, S., Gehringer, E., and Williams, L. (2004). "Sangam: a distributed pair programming plug-in for Eclipse". In Proceedings of the 2004 OOPSLA Workshop on Eclipse Technology Exchange. eclipse '04. ACM Press, New York, NY, 73-77.

Kircher, M., Jain, P., Corsaro, A., Levine, D. (2001) Distributed extreme Programming. In Marchesi and Succi, pages 66-71.

Primo, Alex F. Teixeira, CASSOL, Márcio Borges Fortes. (1999) Explorando o conceito de Interatividade: definições e taxionomias. In: Informática na Educação: teoria e prática. Porto Alegre: Revista PGIE/UFRGS, v.2, n.2, out.

Schummer, T., Schummer J. (2001) Support for Distributed Teams in Extreme Programming. Extreme Programming Examined, Succi, G., Marchesi, M. eds., p. 355–377, Boston, MA: Addison Wesley.

Stotts, David, Laurie Williams, Nachiappan Nagappan, Prashant Beheti, Dennis Jen, and Anne Jackson (2003) "Virtual Teaming: Experiments and Experiences with Distributed Pair Programming", Proceedings of the Third XP Agile Universe Conference, 2003.

Williams, L. A., Kessler, R. R. (2000). All I really need to know about pair programming I learned in kindergarten. ACM Communications. Vol. 43, No. 5, 108-114.