Alterando o Grau de Colaboração de uma Ferramenta de Compartilhamento de Desktop por Meio de Plugins

Eduardo Telles Carlos
Tecgraf / DI / PUC-Rio
R. M. S. Vicente, 225
22453-900 – Rio de Janeiro, RJ
+55 21 2512-5984 - Brasil
etc@tecgraf.puc-rio.br

Alberto Raposo
Tecgraf / DI / PUC-Rio
R. M. S. Vicente, 225
22453-900 – Rio de Janeiro, RJ
+55 21 2512-5984 - Brasil
abraposo@tecgraf.puc-rio.br

RESUMO
Este artigo apresenta a experiência de desenvolvimento do VDTool (*Virtual Desktop Tool*), uma ferramenta para auxiliar processos colaborativos entre dois participantes, permitindo a troca de informações via captura e envio de tela. A aplicação surgiu da necessidade de simplificação de uma ferramenta de videoconferência existente. A partir de uma versão inicial, foram desenvolvidos *plugins* para incluir novos recursos de cooperação e coordenação na aplicação. O modelo de colaboração 3C serviu como mecanismo de análise para este desenvolvimento.

ABSTRACT
This paper presents the experience of developing VDTool (Virtual Desktop Tool), a tool to support collaborative processes between two participants, providing information exchange by means of screen capturing and transmitting. The application appeared as a consequence of the necessity of simplifying an existing videoconferencing tool. On top of an initial version, plugins were developed in order to add new cooperation and coordination facilities to the application. The 3C collaboration model was used as an analysis tool in this development.

Palavras-Chave
Colaboração, Modelo de Colaboração 3C, Plugins.

Keywords
Collaboration, 3C Collaboration Model, Plugins.

1. INTRODUÇÃO
Atualmente, com o crescente aumento da capacidade de processamento dos equipamentos e do desempenho da comunicação pelas redes, está se tornando cada vez mais viável, especialmente nas grandes companhias, o uso de ferramentas colaborativas. Tais ferramentas têm sido usadas para os mais variados propósitos, tais como realização de reuniões à distância e realização de trabalhos entre especialistas remotamente localizados.

Atendendo a essas necessidades, este trabalho iniciou-se com o desenvolvimento do CSVTool (*Collaboration Supported by Video Tool*), uma ferramenta para colaboração baseada em vídeo, projetada para prover recursos de colaboração para aplicações com nenhum ou pouco suporte à colaboração [1]. O projeto dessa ferramenta foi guiado pelas necessidades de uma grande companhia brasileira, tais como independência de plataforma, simplicidade de uso e suporte a múltiplos usuários sobre rede sem suporte a multicast.

O CSVTool, assim como qualquer outro sistema colaborativo, pode ser analisado de acordo com o modelo 3C de colaboração [2]. Este modelo, inicialmente proposto por Ellis [3], entende que o suporte computacional à colaboração pode ser realizado por meio da interação entre mecanismos de comunicação, coordenação e cooperação. Comunicação está relacionada à troca de mensagens e informações entre as pessoas. Coordenação diz respeito ao gerenciamento de pessoas, suas atividades e recursos. Cooperação, por sua vez, se relaciona à produção que ocorre em um espaço de trabalho compartilhado.

Apesar da separação dos 3Cs para fins de análise, comunicação, coordenação e cooperação não podem ser vistas de forma isolada, uma vez que existe uma constante interação entre elas [4]. Essa relação entre os 3Cs do modelo pode ser usada para guiar o projeto de uma ferramenta colaborativa qualquer, mesmo que ela esteja mais voltada para um dos 3Cs. Uma ferramenta de *chat*, por exemplo, que é uma ferramenta de comunicação, requer comunicação (troca de mensagens), coordenação (políticas de acesso) e cooperação (área de compartilhamento e registro das sessões).

Da mesma forma, o CSVTool, que também é mais voltado para a comunicação, possui não apenas recursos de comunicação (troca de informações por áudio, vídeo e texto), mas também de coordenação (controle de sessão, controles individualizados da recepção e envio de áudio/vídeo para cada participante) e cooperação (transmissão da área de trabalho – *desktop*).

O principal objetivo deste artigo é mostrar como a inclusão de novos "Cs" em uma aplicação colaborativa pode alterar seu grau de colaboração. Para isso, será apresentado um caso de estudo com o VDTool (*Virtual Desktop Tool*), uma ferramenta para auxiliar processos colaborativos entre dois participantes, permitindo a troca de informações pelo envio do *desktop* (via

SBSC Simpósio Brasileiro de Sistemas Colaborativos
20 a 22 de Novembro de 2006
Natal, RN, Brasil

Anais do III Simpósio Brasileiro de Sistemas Colaborativos, pp. 140-149.

captura de tela) e por uma ferramenta de bate-papo. Foram desenvolvidos *plugins* que implementam serviços de alguns "Cs", aumentando o grau de colaboração oferecido pela ferramenta.

Um segundo objetivo do artigo é mostrar que, seguindo o modelo 3C, o caminho nem sempre é unidirecional, i.e., nem sempre o que ser quer é acrescentar novos "Cs". Em alguns casos, as necessidades levam à retirada de "Cs" da ferramenta. Este caso é ilustrado na próxima seção, onde é explicado como surgiu o VDTool, que é uma versão simplificada do CSVTool. A Seção 3 apresenta trabalhos relacionados e a Seção 4 retoma a direção da inclusão de novos "Cs" no VDTool, apresentando os *plugins* desenvolvidos. A Seção 5 apresenta as conclusões e trabalhos futuros.

2. DO CSVTOOL AO VDTOOL

O CSVTool é uma ferramenta de videoconferência baseada no modelo de cliente/servidor, onde o servidor é responsável pelo gerenciamento dos clientes presentes em uma determinada sessão de colaboração. Os clientes, por sua vez, comunicam entre si através de áudio, vídeo e texto (via mensagens instantâneas). Além desses elementos de comunicação, a ferramenta também provê a funcionalidade de envio de *desktop* para visualização. Toda a interação é configurável, i.e., o participante tem controle sobre a comunicação, podendo escolher entre o envio/recebimento de áudio ou vídeo, ou ambos ao mesmo tempo.

Mesmo sendo uma ferramenta prioritariamente voltada para comunicação, o CSVTool, apresenta elementos de coordenação e cooperação enquadrando-se assim no modelo 3C [2]. A figura 1 mostra a decomposição das funcionalidades do CSVTool de acordo com o modelo 3C.

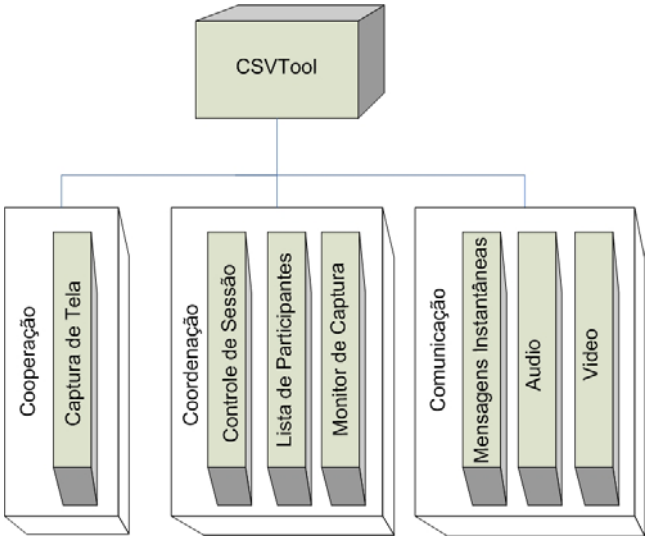

Figura 1- CSVTool e o modelo 3C

A separação das funcionalidades da ferramenta para análise trouxe uma nova visão para a solução de problemas de usabilidade e de implementação, mesmo tendo em mente que os "Cs" têm que interagir entre si, cada qual dando suporte para outro com a finalidade de aumentar o grau de colaboração.

O CSVTool vem sendo usado na empresa que financiou seu desenvolvimento. O processo de adoção de sistemas colaborativos em organizações é um processo muito delicado. Se o software for vendido da maneira usual, "de prateleira", ele pode não ser bem aceito na organização [5]. Na verdade, o processo de adoção de groupware em organizações é um processo dual, tanto de adaptação da organização de trabalho às condições da ferramenta, quanto de adaptação da ferramenta à organização do trabalho na companhia [6, 7]. Apesar de não discordarmos desta afirmação, acreditamos que as chances de sucesso na adoção do software serão maiores se o sistema colaborativo se adaptar completamente à real organização do trabalho na companhia. Por esta razão, o desenvolvimento do CSVTool tem apresentado uma natureza bastante dinâmica e até certo ponto imprevisível, sendo guiado pelas necessidades demandadas pelo seu uso no "mundo real".

O surgimento do VDTool é um exemplo da natureza dinâmica do processo de desenvolvimento do CSVTool. Com o uso do CSVTool na empresa, o recurso de envio de tela despertou o interesse de seus usuários para suprir um antigo problema. Em muitas situações, tais como treinamento e ajuda online, especialistas precisam demonstrar, de maneira mais clara, o que acontece em seus *desktops*, sem a necessidade de um encontro presencial. Nesses casos, a transmissão de *desktop* é um recurso mais importante que a transmissão de áudio e vídeo, visto que os especialistas geralmente se falam ao telefone durante essas situações. Esses usuários viram no CSVTool uma possibilidade para terem seus problemas resolvidos. Porém, alegaram que não queriam complexidades do sistema de videoconferência, tais como transmissão de áudio e vídeo (queriam continuar falando ao telefone) e necessidade de servidor (a colaboração é sempre entre duas pessoas apenas).

Como este não era o foco do CSVTool, houve a necessidade de criação de uma nova ferramenta com algumas características da original, porém menos complexa e que resolvesse o problema em questão.

Iniciou-se assim o desenvolvimento do VDTool, como uma versão simplificada do CSVTool, sem suporte a áudio, vídeo e sem servidor. Atualmente, a ferramenta VDTool auxilia processos colaborativos entre dois participantes, permitindo a troca de informações pelo envio do *desktop* (via captura de tela) e por uma ferramenta de bate-papo. Voltando à figura 1, o VDTool pode ser analisado à luz do modelo 3C como sendo apenas o serviço de cooperação do CSVTool (captura e envio de tela), acrescido de um serviço de coordenação (monitor de captura) e um serviço de comunicação (mensagem instantânea). Os demais serviços de comunicação foram eliminados (áudio e vídeo), bem como os demais serviços de coordenação (controle de sessão e lista de participantes, já que a comunicação no VDTool é um-para-um).

Na figura 2, de cima para baixo e da esquerda para direita, temos a tela remota recebida pelo usuário, a informação da área que está sendo enviada, a janela de eventos executados e, por fim, a troca de mensagens textuais do VDTool. Nesse caso, o usuário está enviando seu *desktop* e também recebendo o *desktop* remoto, de modo que existe tanto a tela remota recebida quanto o monitor de captura, onde ele vê a área da sua tela que está sendo enviada.

No momento, o VDTool tem sido usado com sucesso pelo grupo de usuários que solicitou seu desenvolvimento. Uma das suas características principais é ser multi-plataforma, atualmente disponível nas plataformas Windows, Linux, Solaris e Irix. Porém, a rigor, o VDTool tem poucos recursos de colaboração

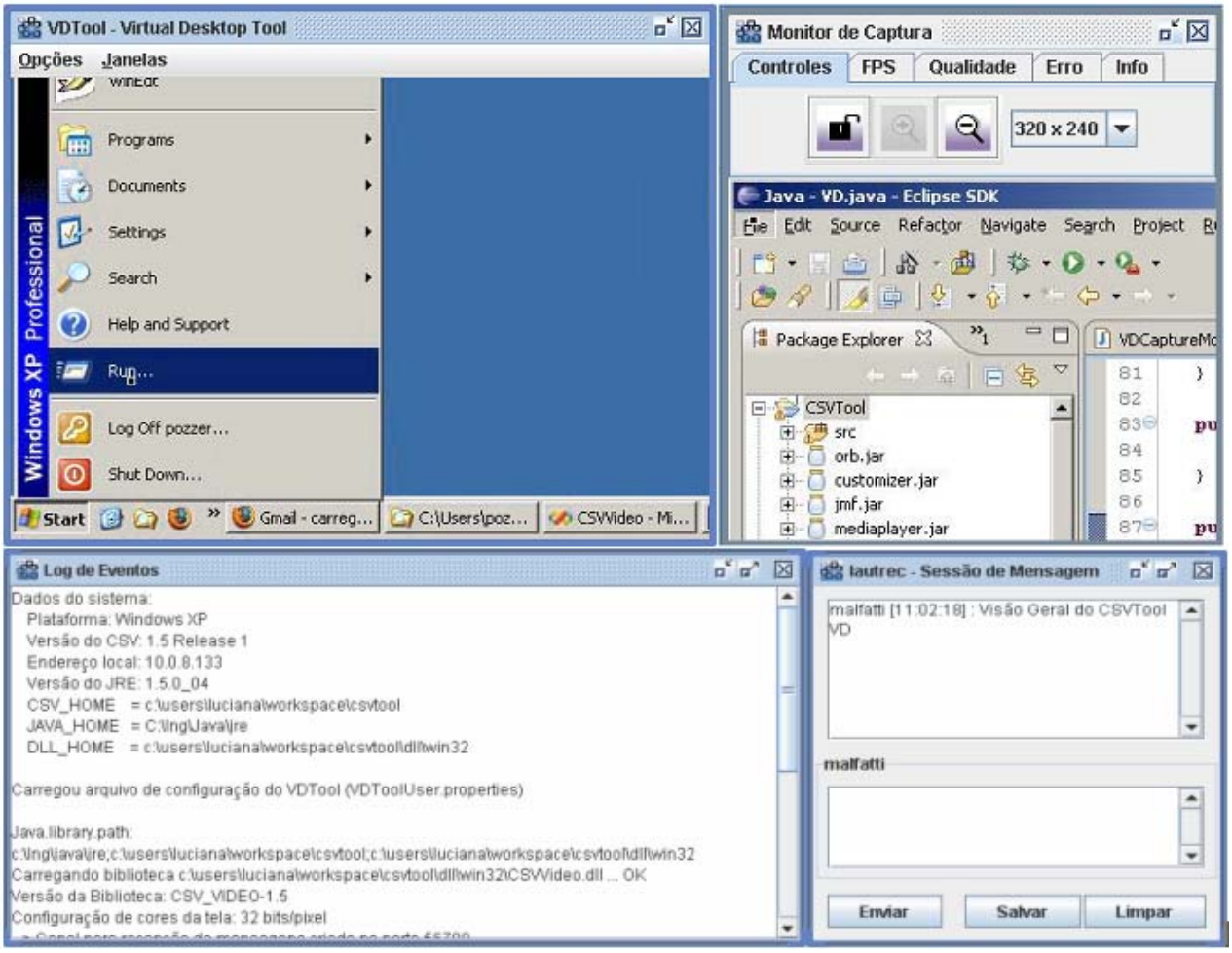

Figura 2 – Visão geral do VDTool

(i.e., poucos "Cs"). Com o intuito de aumentar o grau de colaboração do VDTool, foram inseridos *plugins* colaborativos na ferramenta.

A adição dos *plugins* trouxe novas funcionalidades para o sistema, acrescentando novos "Cs" à ferramenta e transformando-a novamente em uma ferramenta com bom grau de colaboração, embora com propósito totalmente diferente do CSVTool original. A figura 3 mostra a decomposição das funcionalidades do VDTool em relação ao modelo 3C. Nessa figura, os serviços que aparecem com fundo branco foram herdados do CSVTool, enquanto os que aparecem com o fundo colorido foram implementados pelos *plugins*.

A próxima seção esclarece a idéia dos *plugins* colaborativos como parte integrante de ferramentas colaborativas.

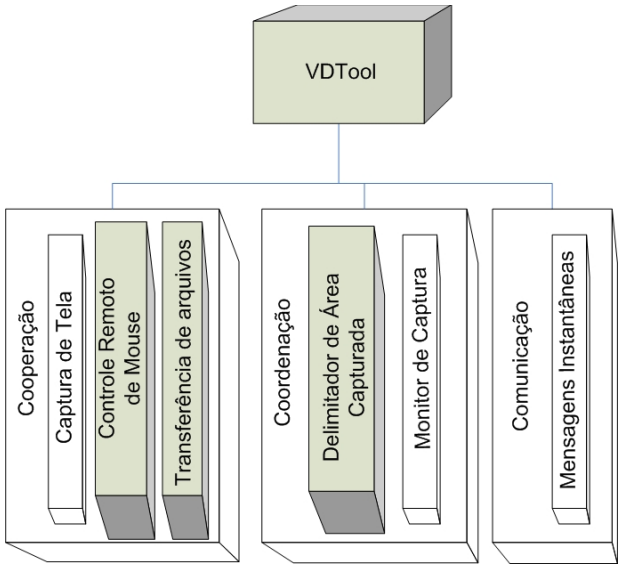

Figura 3-VDTool e o modelo 3C

3. TRABALHOS RELACIONADOS

Uma vez que o VDTool já atendia os requisitos solicitados, o próximo passo foi voltar a acrescentar novos "Cs", tornando-a uma ferramenta com alto grau de colaboração, embora com propósito diferente do CSVTool.

Experiências similares, seguindo o modelo 3C, foram realizadas em [8,9], onde foram adicionados novos serviços de coordenação e cooperação a ferramentas de comunicação do ambiente AulaNet. A diferença em termos de implementação é que nos casos acima, foram adicionados componentes de software integrados a um *framework* de componentes. No caso do VDTool, como a aplicação não foi desenvolvida com o paradigma de componentes, a solução adotada para o acréscimo de novas funcionalidades foi o desenvolvimento de *plugins*.

Um *plugin* é um programa que deve interagir com outro programa para prover alguma funcionalidade particular [10]. O software principal determina a forma pela qual os *plugins* são incorporados ao sistema e a maneira que irão trocar informações. Uma das grandes vantagens do uso de *plugins* como parte integrante de um sistema é a não necessidade de recompilação ou substituição de partes do sistema original, o que pode gerar problemas de versão. Na maioria das vezes, os sistemas que dão suporte a *plugins* disponibilizam uma API que possibilita a interação entre eles. Os *plugins*, por sua vez, são implementados na forma de bibliotecas dinâmicas. Atualmente, vários programas profissionais disponibilizam APIs para desenvolvedores estenderem as funcionalidades de seus sistemas, aumentando assim o tempo até que seus sistemas se tornem obsoletos.

Na fase anterior ao desenvolvimento dos *plugins*, foi feita uma análise de sistemas colaborativos similares ao VDTool, explorando seus pontos fortes e fracos. A partir desta análise, foi possível o desenvolvimento voltado para implementar no VDTool algumas características interessantes encontradas nos sistemas analisados e também para suprir algumas limitações encontradas nos mesmos.

O VRVS (Virtual Room Videoconferencing System) [11] foi desenvolvido no início de 1997 e inicialmente usado dentro da comunidade HENP (High Energy and Nuclear Physics). Recentemente, o serviço foi estendido para outros grupos acadêmicos e de pesquisa. O VRVS é um sistema orientado para a *web*, para videoconferência e trabalho colaborativo. O desenvolvimento do sistema inclui hoje milhares de *hosts* registrados rodando o software em mais de 60 países. O VRVS fornece várias funcionalidades colaborativas como: envio e recebimento de áudio e vídeo, compartilhamento de *desktop* e aplicações e *chat* em várias plataformas. A análise deste software nos mostrou a dependência de outros sistemas para o funcionamento, como o MS Internet Explorer e o WinVNC, o que se torna uma dificuldade a mais para o usuário. Além disso, a existência de videoconferência, o coloca mais próximo do CSVTool do que do VDTool, que precisa manter sua simplicidade.

O RealVNC [12] foi desenvolvido com o intuito de melhorar o design e a implementação do VNC (Virtual Network Computing) [13]. O VNC é um sistema de compartilhamento de *desktop* para controlar outros computadores remotamente. Além do RealVNC, existem outras variantes desse sistema, como o TightVNC e o UltraVNC, que também foram analisadas para este trabalho. Essas ferramentas têm maior grau de colaboração que o VDTool em seu estágio antes dos *plugins*, visto que ele ainda não permitia o acesso ao *desktop* remoto, mas apenas sua visualização. Por esta razão, algumas das funcionalidades dos "VNCs" foram analisadas para serem incorporadas ao VDTool na forma de *plugins*. Algumas deficiências foram encontradas na parte de usabilidade, como por exemplo, a ausência de mecanismos de coordenação para evitar "brigas" pelo mouse controlado local e remotamente, e a ausência de algoritmos de escala na imagem recebida, assuntos que serão discutidos na Seção 4.1.

Outra ferramenta relacionada é o Sun SharedShell [14], que permitia o compartilhamento de um terminal de comando Unix (*shell*) entre usuários remotamente localizados, acrescentando também recursos de *whiteboard* compartilhado e gravação da sessão colaborativa. Seu intuito era permitir que o responsável pelo suporte Unix visse e interagisse com o terminal de comando do usuário. Além de restrita a Unix, essa ferramenta parece ter sido descontinuada.

4. PLUGINS COLABORATIVOS

O desenvolvimento dos *plugins* para o VDTool foi baseado na idéia do modelo 3C, i.e., cada *plugin* dando o suporte necessário para cada um dos "Cs" acrescentado, com a finalidade de aumentar assim o grau de colaboração. Uma vez implantados, os *plugins* têm que manter a coesão entre todos os "Cs" e, quando incorporados a um determinado sistema colaborativo, não fugir ao foco principal pelo qual a ferramenta original foi criada.

Seguindo esta idéia, são apresentados a seguir os *plugins* desenvolvidos e incorporados na ferramenta VDTool.

4.1 Controle Remoto de Mouse

O VDTool, por ser uma ferramenta colaborativa, deve prover o máximo de interação entre os participantes. Antes do desenvolvimento do *plugin* em questão, esta ferramenta não possuía nenhum esquema de acesso remoto, ou seja, não era possível compartilhar a aplicação remotamente, apenas visualizá-la. Este foi um dos problemas encontrados que diminuía a interação entre os usuários.

A idéia do *plugin* de controle remoto de mouse é disponibilizar a funcionalidade de controlar uma máquina remota através do mouse no mesmo ambiente do sistema colaborativo que incorporará este *plugin*. A finalidade principal do *plugin* é poder ampliar a capacidade do espaço de trabalho compartilhado, sendo portanto um *plugin* de cooperação.

A figura 4 mostra os processos envolvidos desde o início da execução do VDTool até a chamada do *plugin* de controle remoto de mouse.

Assim como a maioria dos *plugins* de outros sistemas, este foi implementado seguindo a estrutura de biblioteca dinâmica em Linguagem C. Como o VDTool foi desenvolvido na Linguagem Java, a incorporação deste utilizou a interface de comunicação JNI (Java Native Interface) [15]. Com isso, ocorre uma constante troca de informações entre o *plugin* e o VDTool.

Para o desenvolvimento deste *plugin*, houve o cuidado de se manter a característica de ser multi-plataforma, uma exigência da empresa que solicitou o desenvolvimento. Para tal, o código possui desvios condicionais destinados a cada plataforma.

A primeira informação a ser obtida para o funcionamento do *plugin* de controle remoto é a determinação da posição do mouse no sistema, para que ela possa ser enviada e posteriormente reconhecida na máquina remota (i.e., a máquina que está tendo seu *desktop* controlado). Neste ponto, definimos que uma posição é válida para ser enviada se o mouse estiver dentro da janela da

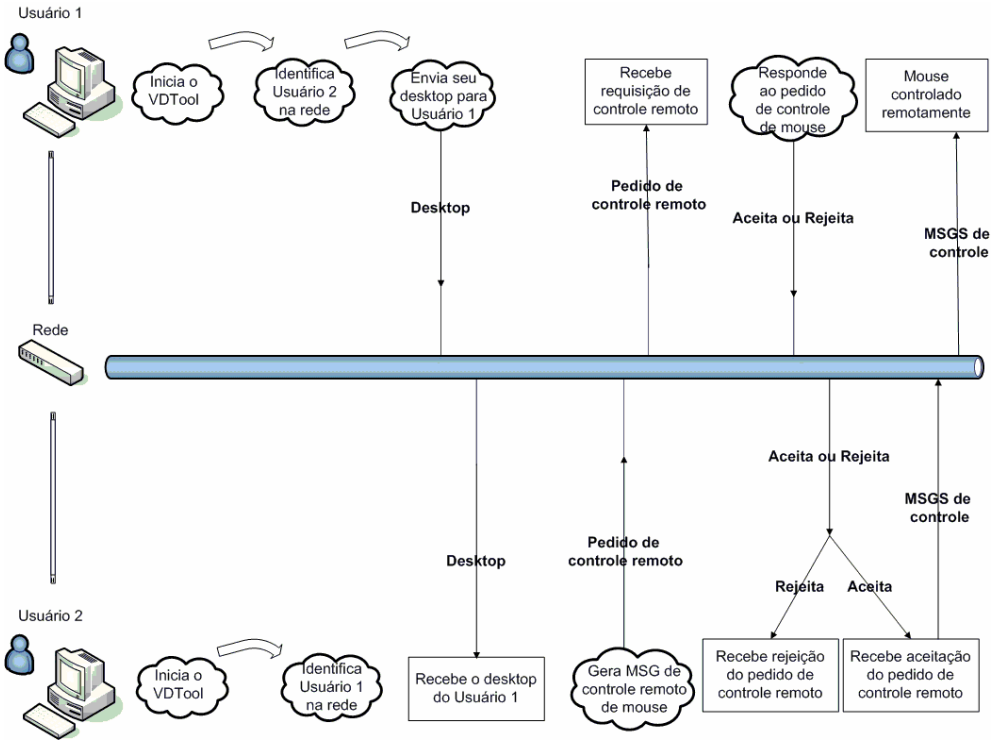

Figura 4-Diagrama do processo de acesso remoto

aplicação e sua coordenada seja diferente da última capturada, como mostrado na figura 5. Analogamente à posição do mouse, capturamos também os eventos referentes ao clique e *drag* do mouse para serem enviados caso a posição seja válida. Essas informações são então transmitidas via rede para a máquina remota.

A posição do mouse que chega à máquina remota não representa a posição real, em virtude da escala que a imagem sofreu na máquina local (figura 5), onde o usuário recebe toda a área de trabalho da máquina remota em uma imagem escalada (a escolha de um algoritmo ótimo para escalar imagens será explorado na Seção 4.1.1). Com isso a posição deve ser corrigida seguindo a seguinte fórmula:

$$Posição\_real = \left(\frac{posição\_na\_imagem\_escalada * dimensão\_do\_desktop}{zoom} \right)$$

A partir do momento que temos a posição real do mouse (calculada pela fórmula acima, a partir da posição enviada através da rede pelo usuário local) na máquina remota e um possível evento, podemos simular esta ação utilizando funções nativas de cada plataforma.

Neste ponto, o mouse está recebendo informações tanto do usuário remoto quanto do usuário local, podendo ocorrer uma "briga" pelo mesmo se os dois mandarem informações diferentes ao mesmo tempo. Para minimizar este problema, duas abordagens foram desenvolvidas. A primeira tenta explorar a idéia de *awareness* dando um *feedback* ao usuário remoto, informando que seu mouse está sendo controlado pelo outro usuário. Este *feedback* é dado pela alteração momentânea da aparência do ponteiro do mouse na máquina controlada (foi utilizado um cursor animado). Ao fim do controle remoto, as propriedades anteriores do ponteiro são restabelecidas.

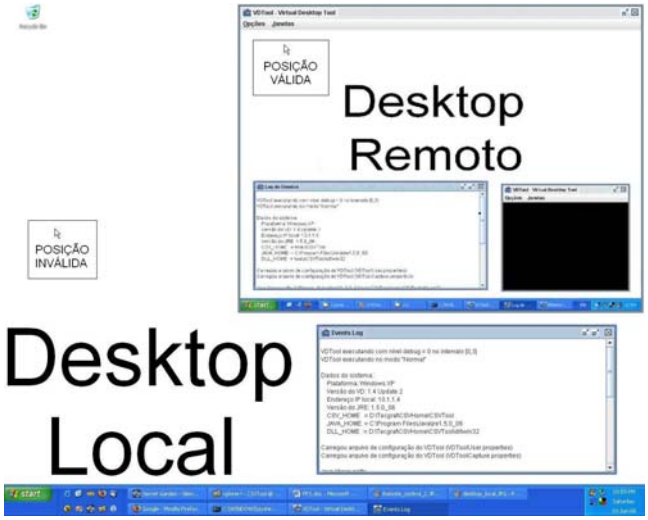

Figura 5-Posição válida do mouse (para ser enviada para a máquina remota)

Para tentar diminuir mais a possibilidade de "briga" pelo mouse, foi desenvolvida uma segunda abordagem que explora a idéia de um segundo ponteiro de mouse para o usuário remoto. A literatura não disponibiliza muito material para esta abordagem, contudo, o desenvolvimento do mesmo foi estudado e duas soluções surgiram: a primeira foi a simulação de um *driver* virtual USB que receberia eventos do usuário remoto e responderia às suas

ações [16] e a segunda, o desenho de um ponteiro "virtual" utilizando primitivas de desenho de cada plataforma e compartilhamento dos eventos de clique com o ponteiro real.

Neste trabalho apenas a segunda idéia foi implementada. Porém, ao fim do desenvolvimento, verificou-se que os resultados não foram satisfatórios, pois a operação de *redraw* das primitivas nativas de desenho do ponteiro não se mostrou eficiente para aplicações síncronas. Além disso, como os eventos de clique e *drag* do segundo ponteiro eram compartilhados com o ponteiro real, situações de conflito continuaram a ocorrer na geração desses eventos.

Por motivos de segurança, o cancelamento do controle remoto na máquina controlada é feito através da tecla *esc*, uma vez que poderia ocorrer "briga" pelo mouse neste momento.

4.1.1 Algoritmo de Escala

A necessidade do algoritmo de escala surgiu para solucionar dois problemas. O primeiro é que à medida que o usuário decide enviar uma parcela maior do seu *desktop* para o outro usuário, a área capturada que aparece no monitor de captura do VDTool teria que ser ampliada, aumentando assim o monitor de captura e o tráfego de dados na rede, o que não é desejado em algumas situações. O segundo problema é a necessidade de *scroll* na imagem recebida pela máquina que está realizando o controle remoto de mouse, problema que encontramos em ferramentas de acesso remoto, como os "VNCs" analisados.

Para a escolha do melhor algoritmo de escala de imagem para ferramentas síncronas foi levado em conta o que tivesse a melhor relação qualidade *vs*. desempenho. Foram estudados vários algoritmos presentes na literatura. Tal estudo identificou que o algoritmo dct, implementado em C, apresentou melhor qualidade [17]. Porém, é necessário levar em conta também o tempo de processamento do algoritmo. O algoritmo de escala deve ser rápido o suficiente para não se tornar o "gargalo" da aplicação e ter resultados satisfatórios. Com isso, os algoritmos disponíveis na Linguagem Java foram descartados, pois têm o tempo de processamento elevado em virtude da própria linguagem. O dct, apesar de sua melhor qualidade, também não obteve um desempenho muito bom. Ao final do estudo, chegou-se à conclusão que o algoritmo de *downsize rought* [18] obteve a melhor relação qualidade *vs.* desempenho, sendo por este motivo escolhido para ser incluído no *plugin* de controle remoto de mouse. A idéia do algoritmo de *downsize rought* é fazer uma média de cada componente RGB presente na imagem de acordo com um fator entre o tamanho da imagem original e o tamanho da imagem final como pode ser visto na figura 6. No caso mostrado na figura, a imagem original tem suas dimensões reduzidas à metade. Nesse caso, cada *pixel* da imagem reduzida é obtido como uma média de uma área de 2x2 *pixels* da imagem original.

4.2 Delimitador de Área Capturada

Para tornar mais claro ao usuário o que está sendo capturado e enviado da sua máquina para a outra, o VDTool utiliza o monitor de captura, ilustrado na figura 2. Quando ativo, além de dar o *feedback* necessário ao usuário, provê informações sobre o estado atual do sistema, como a banda passante e também disponibiliza opções para configuração do sistema, como a qualidade da imagem enviada. Porém, sua permanência na área de trabalho gera poluição visual, uma vez que o usuário que está enviando seu *desktop* perde área na tela, tendo assim que deslocar esta janela sempre que necessário para desempenhar outras atividades. Este foi mais um dos problemas identificados na ferramenta VDTool, que acabam por atrapalhar a colaboração entre os usuários.

	1	2	3	4
1	R = 255 G = 0 B = 0	R = 197 G = 208 B = 48	R = 13 G = 13 B = 45	R = 255 G = 16 B = 78
2	R = 153 G = 153 B = 255	R = 109 G = 94 B = 29	R = 0 G = 126 B = 244	R = 41 G = 0 B = 10
3	R = 77 G = 102 B = 0	R = 255 G = 255 B = 0	R = 10 G = 154 B = 247	R = 148 G = 199 B = 45
4	R = 33 G = 48 B = 204	R = 0 G = 0 B = 255	R = 47 G = 255 B = 255	R = 255 G = 255 B = 255

	1	2
1	R = 178 G = 114 B = 83	R = 77 G = 39 B = 94
2	R = 91 G = 101 B = 115	R = 115 G = 216 B = 200

Figura 6-Ilustração do algoritmo de escala *downsize rought*

Algumas soluções foram levantadas, como a diminuição da área ocupada pelo monitor de captura no *desktop*. Porém, isto ainda causaria certo incômodo e perderíamos o poder de *awareness* dos elementos da interface gráfica. Outra solução estudada foi fazer o uso de transparência em cima da janela do VDTool, porém isto iria impossibilitar o usuário de acessar possíveis elementos que estejam embaixo da janela. A melhor alternativa encontrada para tal problema foi o desenvolvimento de um *plugin* que delimita a área capturada através do desenho de linhas no próprio *framebuffer* do usuário que, ao mesmo tempo em que forneça o *feedback* ao usuário, permita que o mesmo minimize o monitor de captura quando necessário e continue trabalhando com toda a área de trabalho. Por ter o intuito de ajudar no gerenciamento da interação entre os usuários e ter funcionalidade complementar à do monitor de captura, este *plugin* é considerado um serviço de coordenação.

Assim como o *plugin* de controle remoto de *mouse*, este foi implementado em forma de biblioteca dinâmica na Linguagem C e com diretivas condicionais de compilação para cada plataforma. A figura 7 mostra a troca de informações entre o VDTool e o *plugin* de delimitação de área capturada.

Como o delimitador de área capturada foi implementado utilizando código nativo na Linguagem C, houve mais uma vez a necessidade da utilização da JNI para que fosse possível a comunicação entre o VDTool e o *plugin*. Nesse ponto houve alguns problemas relativos à plataforma Unix e à posição do ponteiro do mouse nas extremidades da tela. No *plugin* delimitador de área capturada, as rotinas de desenho no *framebuffer* eram chamadas e, posteriormente, o código em Java era chamado novamente. Este esquema foi válido para a plataforma Windows, porém inválido para as plataformas Unix, uma vez que nesta era impossível sair do *loop* de *redraw* das rotinas de desenho no *framebuffer* e retornar ao código Java. Para resolver este problema, foram criados executáveis para serem chamados a partir do código em Java e executados em paralelo com o código em Java do VDTool. Dessa forma, esses executáveis desempenham a mesma funcionalidade do *plugin* desenvolvido para a plataforma Windows.

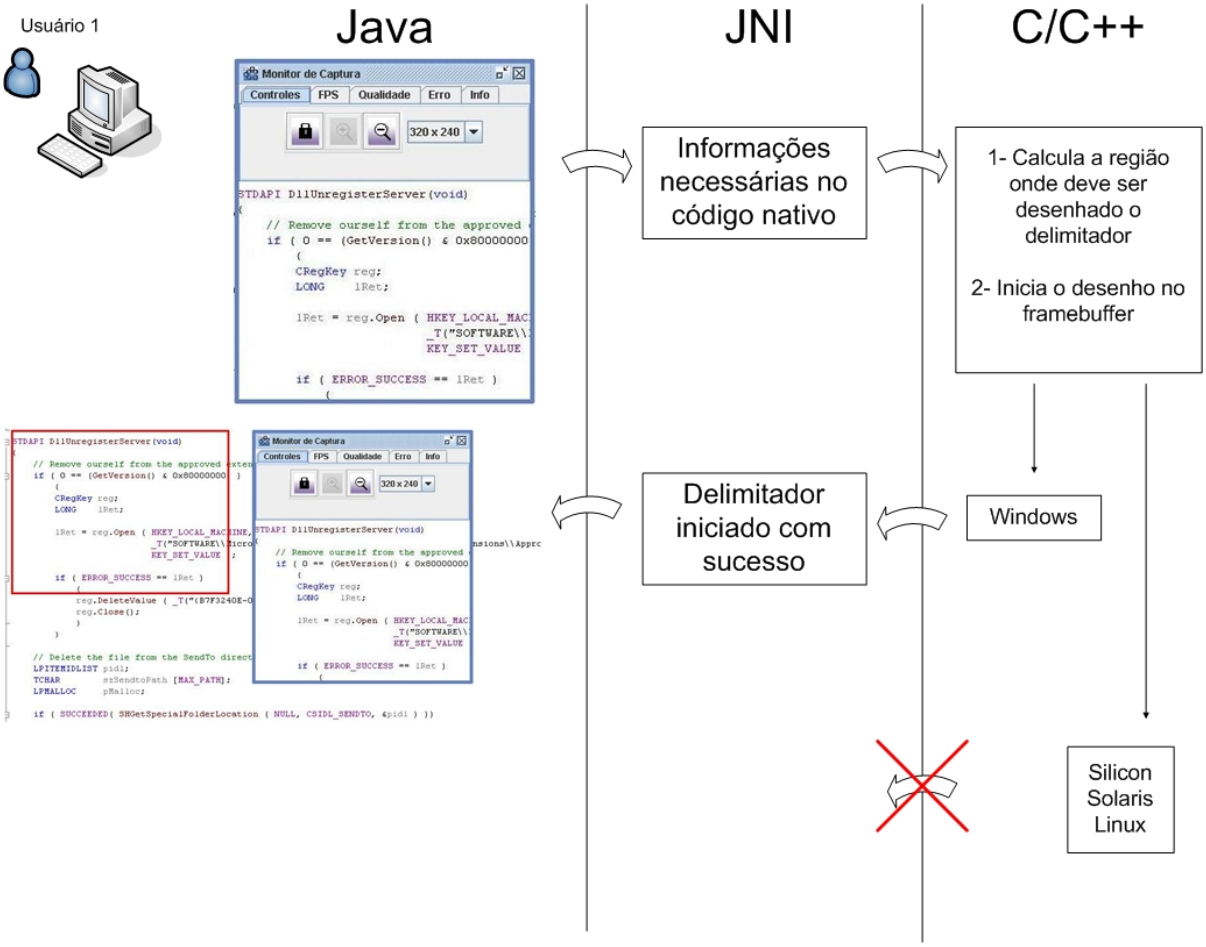

Figura 7 – Delimitador de área capturada e o VDTool

4.3 Transferência de Arquivos

A transferência de arquivos entre usuários é uma funcionalidade muito comum e útil em ferramentas colaborativas. A maior motivação para o desenvolvimento desse *plugin* foi a ausência desta funcionalidade no VDTool e a presença na maioria das ferramentas analisadas anteriormente. Porém, em algumas dessas ferramentas, a transferência de arquivos ocorre em apenas um sentido, i.e., sendo permitido apenas o envio de arquivos da máquina que está controlando a outra remotamente (*upload* de arquivos). O RealVNC, por exemplo, faz a transferência de arquivos bidirecional em sua versão comercial, mas não faz nenhum tipo de transferência em sua versão gratuita.

Assim como no caso do controle remoto do mouse, o *plugin* de transferência de arquivos amplia a capacidade do espaço de trabalho compartilhado, se enquadrando como um serviço cooperação no modelo 3C.

A maioria das aplicações colaborativas que disponibilizam a funcionalidade de transferência de arquivos utiliza a própria janela da aplicação para realizar essa transferência, i.e., a transferência é feita a partir de um elemento da interface do programa. Como um dos focos deste trabalho é o reuso de *plugins* colaborativos em diversas ferramentas da forma mais transparente possível, tanto para o desenvolvedor quanto para o usuário final, essa abordagem de usar a própria janela da aplicação não foi utilizada.

Para implementação deste *plugin* foram estudadas duas opções: a primeira fazendo a transferência de arquivos entre usuários utilizando *drag and drop* e a segunda fazendo a transferência através do *context menu* do Windows.

A idéia de transferência via *drag and drop* pode ser vista de duas formas. A primeira forma é para realizar o *upload* de arquivos ou pastas entre o sistema da máquina local (*drag*) para a janela do VDTool, representando a máquina remota (*drop*), como pode ser visto na figura 8. A segunda forma é para realizar o *download* de arquivos ou pastas da máquina remota através da janela do VDTool (*drag*) para a máquina local (*drop*).

Inicialmente, esta seria a única forma de transferência de arquivos implementada na forma de *plugins*, porém houve a necessidade de termos outra forma de transferência devido a problemas encontrados em testes na parte referente à execução de *download* de arquivos ou pastas da máquina remota para a máquina local. A primeira etapa para realizar a transferência de dados entre a máquina remota e a máquina local via *drag and drop* é a identificação da ação de *drag* de algum arquivo ou pasta, ou um conjunto deles. Identificado este evento, executado pelo usuário

que está comandando remotamente a outra máquina, a próxima etapa é disparar o evento de transferência dos elementos que estejam sofrendo *drag*. O início deste evento só será efetivado caso os elementos que estejam sendo "arrastados" passem por uma das bordas do *desktop* da máquina controlada ilustrado em vermelho na figura 9.

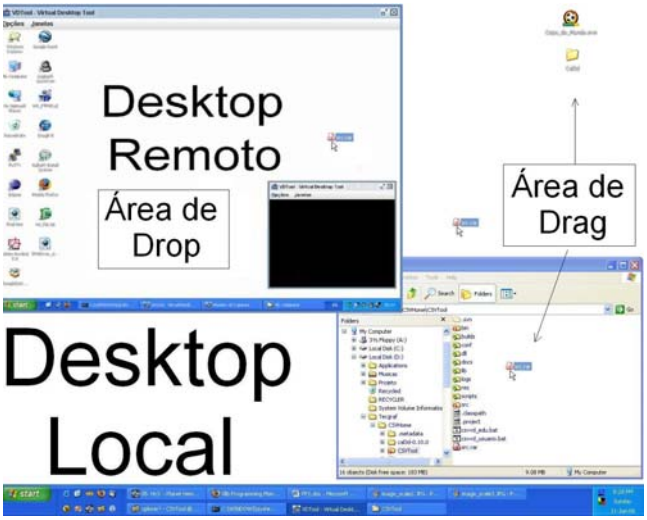

Figura 8-Áreas de *drag and drop* para *upload* de arquivos

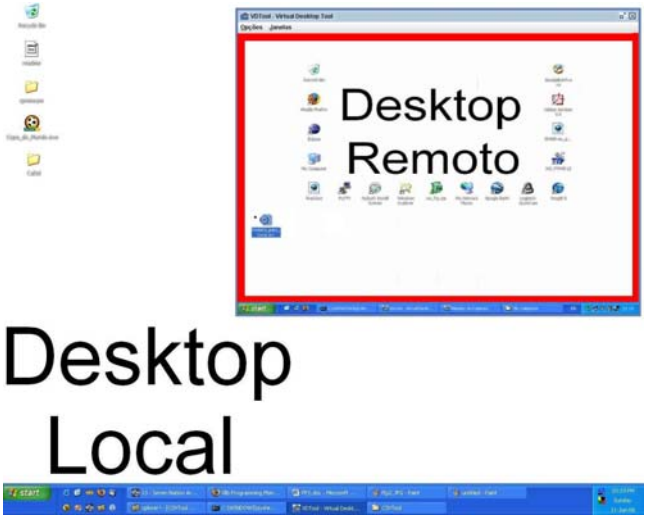

Figura 9-Área de inicio do *download* em vermelho

Ao entrar na área delimitada, o *plugin* identifica o caminho para todos os elementos que estão sofrendo *drag* para que posteriormente possa iniciar a transferência dos mesmos para a outra máquina. Neste ponto justifica-se a inclusão de uma maneira alternativa de executar esta funcionalidade, pois nem sempre é possível identificar o caminho para todos os elementos de maneira rápida (necessidade em ferramentas síncronas), uma vez que esta informação é requisitada ao sistema operacional da máquina remota e não é recebida imediatamente. Esta situação não ocorre no *upload*, pois o sistema operacional da máquina remota não precisa ser consultado.

A outra forma de transferência de arquivos implementada para este *plugin* foi a utilização do *context menu* do Windows (figura 10). Desta forma os problemas mencionados anteriormente são contornados, pois a transferência é iniciada imediatamente após a sua solicitação. Como esta parte do *plugin* envolve a inserção de novas entradas no registro do Windows, ao finalizar a conexão entre os participantes, o *plugin* deve ser chamado novamente para retirar a entrada no registro do sistema operacional.

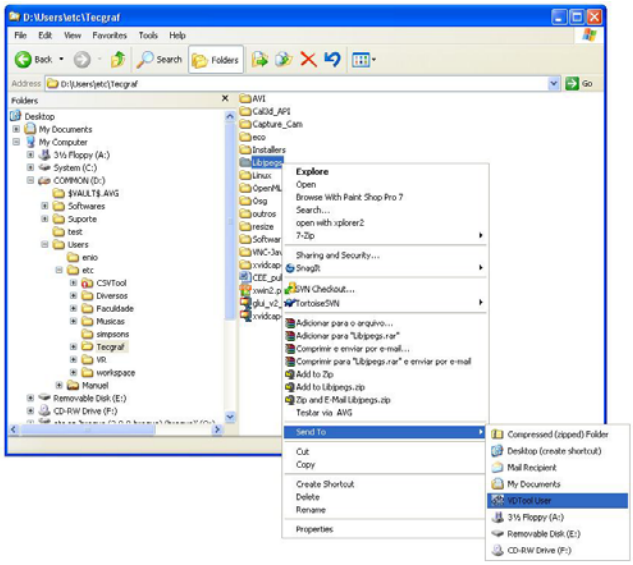

Figura 10-Transferência de arquivos via *context menu*

Diferentemente dos outros *plugins*, este só foi implementado para a plataforma Windows, por ser o sistema operacional mais utilizado e aparentemente o mais simples para o desenvolvimento, em uma primeira análise. Para manter o requisito de portabilidade multi-plaforma, este *plugin* ainda precisa ser implementado nas outras plataformas.

5. CONCLUSÕES

A integração visual entre aplicações executadas em locais distintos permite que profissionais geograficamente distribuídos trabalhem colaborativamente, reduzindo as barreiras de comunicação e aumentando a produtividade. Essa possibilidade tem atraído a atenção das empresas, que cada vez mais buscam sistemas colaborativos adequados às suas necessidades.

Porém, uma vez adotada nas organizações, a tecnologia só se torna efetiva se a ela for atribuída um significado e se ela realmente for utilizada [7]. O exemplo do CSVTool/VDTool mostra que este "significado", do ponto de vista dos usuários na empresa, pode ser surpreendente aos olhos dos desenvolvedores de groupware. Cabe aos desenvolvedores, portanto, buscar o entendimento deste significado e encontrar os meios tecnológicos para implementá-los em suas aplicações.

O modelo 3C, como ferramenta de "engenharia" de domínio, se mostrou eficiente para ajudar a cobrir a distância existente entre os requisitos sociais e a viabilidade tecnológica [19]. O caso do VDTool mostra como a análise, à luz do modelo 3C, revela que a exclusão e a inclusão de diferentes "Cs" podem alterar significativamente a funcionalidade de uma aplicação colaborativa, adequando-a a novos propósitos. Dessa forma, abre-

se caminho para que os sistemas colaborativos se adaptem às necessidades dos usuários, e não o contrário, o que aumenta as chances de sucesso do software.

A maneira utilizada para incluir novas funcionalidades colaborativas no VDTool foi pela implementação de *plugins* que realizem tais funcionalidades. Um dos objetivos do desenvolvimento via *plugins* é o possível reuso dos mesmos em outras aplicações colaborativas além do VDTool. Para isso, seria necessário que a outra aplicação tenha uma API flexível para interagir com os *plugins* desenvolvidos. Uma possibilidade alternativa seria o desenvolvimento baseado em componentes de software, como realizado em outras aplicações [2]. Entretanto, essa opção exigiria uma reimplementação do VDTool e por isso não foi escolhida.

Com relação às outras ferramentas analisadas, pode-se dizer que o VDTool é diferente de todas elas, embora não necessariamente "mais colaborativo". Com relação ao VNC, o VDTool permite a transferência de arquivos via *drag and drop* entre a máquina local e a remota, o que não acontece no VNC, onde o *desktop* da máquina local "desaparece" quando se acessa o *desktop* da máquina remota. Com relação ao Sun SharedShell, o VDTool compartilha todo o *desktop*, e não só uma janela de comandos. Porém, o VDTool não possui o recurso de gravação de sessão do Sun SharedShell. Quanto ao VRVS, que é um software mais abrangente, o VDTool é, por requisito de projeto, mais simples. Porém, uma possível integração do CSVTool com o VDTool poderá se aproximar do VRVS, em termos de recursos de colaboração. Entretanto, independente dos recursos de colaboração, uma inovação do VDTool é o uso de *plugins*, que permitem alterar seu grau de colaboração, inserindo novas funcionalidades.

Voltando a mencionar a dualidade no processo de adoção de uma tecnologia em uma organização, podemos dizer que a transformação do CSVTool no VDTool foi uma adaptação da ferramenta às pessoas. Por outro lado, a implementação dos *plugins* para aumentar o grau de colaboração do VDTool está sendo uma tentativa de oferecer mais recursos aos usuários, que poderão adaptar seus processos de trabalho a estes novos recursos.

Uma das próximas etapas do trabalho é verificar a aceitação desses novos recursos junto aos usuários, bem como realizar experimentos que comprovem a alteração (preferencialmente de forma positiva) no grau de colaboração provido pelos novos recursos do VDTool. Outra atividade futura será a avaliação da possibilidade de incluir os *plugins* do VDTool no CSVTool, tornando este último uma ferramenta de colaboração mais completa, com mais recursos de cooperação principalmente, como a transferência de arquivos e controle remoto de mouse.

6. AGRADECIMENTOS

Agradecemos aos Profs. Marcelo Gattass, coordenador do Tecgraf (Grupo de Tecnologia em Computação Gráfica) e Hugo Fuks, coordenador do Groupware@LES, DI, PUC-Rio, DI, PUC-Rio pelo constante apoio ao desenvolvimento deste trabalho. O Tecgraf é um grupo prioritariamente financiado pela Petrobras.

7. REFERÊNCIAS

[1] Pozzer, C. T., Lima, L. S., Raposo, A. B., Vieira, C. J. G., A Multi-user Videoconference-based Collaboration Tool: Design and Implementation Issues. In *Proceedings of the 9th International Conference on CSCW in Design (CSCWD 2005)* (Coventry, U.K.), 2005, 547-552.

[2] Fuks, H., Raposo, A. B., Gerosa, M.A., Lucena, C. J. P. Applying the 3C Model to Groupware Development. *International Journal of Cooperative Information Systems, 14, 2-3* (Jun.-Sep. 2005), 299-328.

[3] Ellis, C. A., Gibbs, S. J., Rein, G. L. Groupware: Some Issues and Experiences. *Communications of the ACM, 34, 1* (Jan. 1991), 38-58.

[4] Pimentel, M., Fuks, H., Lucena, C.J.P. Mediated Chat 2.0: Embedding Coordination into Chat Tools. In *Conference Supplement of the 6th International Conference on the Design of Cooperative Systems (COOP 2004)* (Amsterdam, The Nederlands), 2004, 99-103.

[5] Grudin, J. Groupware and Social Dynamics: Eight Challenges for Developers. *Communications of the ACM, 37, 1* (Jan. 1994), 92-105.

[6] Bardram, J.E. Organizational Prototyping: Adopting CSCW Applications in Organisations, In G. Mark et al. (organizers), CSCW 96 Workshop: *Introducing groupware in organizations: What leads to successes and failures?* 1996.

[7] Orlikowski, W. J. The Duality of Technology: Rethinking the Concept of Technology in Organizations. *Organization Science, 3, 3* (Ago. 1992), 398-427.

[8] Fuks, H., Raposo, A.B, Gerosa, M.A., Pimentel, M., Lucena, C.J.P. Suporte à Coordenação e à Cooperação em uma Ferramenta de Comunicação Textual Assíncrona: Um Estudo de Caso no Ambiente Aulanet. In *Anais do I Workshop Brasileiro de Tecnologias para Colaboração (WCSCW 2004)* (Ribeirão Preto, Brasil), 2004, 173-180.

[9] Gerosa, M. A., Raposo, A.B., Fuks, H., Lucena, C.J.P. Uma Arquitetura para o Desenvolvimento de Ferramentas Colaborativas para o Ambiente de Aprendizagem AulaNet. In *Anais do Simpósio Brasileiro de Informática na Educação (SBIE 2004)* (Manaus, Brasil), 2004, 168-177.

[10] Plug-in. *Wikipedia*. Acesso em 25 de Julho, 2006, a partir do Answers.com Web site: http://www.answers.com/topic/plugin

[11] Virtual Rooms Videoconferencing System. Acesso em 26 de Julho de 2006: http://www.vrvs.org

[12] RealVNC. Acesso em 26 de Julho de 2006: http://www.realvnc.com

[13] Richardson, T., Stafford-Fraser, Q., Wood, K.R., Hopper, A. Virtual network computing. *IEEE Internet Computing, 2, 1* (Jan.-Fev. 1998), 33-38.

[14] Yankelovich, N., "Bo" Begole, J., Tang, J. C. Sun SharedShell tool. In *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work (CSCW 2000)* (Philadelphia, USA), 2000, 351.

[15] Sun Microsystems, Inc. *JNI (Java Native Interface) Specification*. Acesso em 25 de Julho, 2006: http://java.sun.com/j2se/1.4.2/docs/guide/jni/spec/jniTOC.html

[16] Wierzchowski, G. *Second Mouse in X mini-HOWTO*. Acesso em 26 de Julho de 2006: http://www.faqs.org/docs/Linux-mini/XFree86-Second-Mouse.html

[17] Martucci, S.A. Image resizing in the discrete cosine transform domain. In *Proceedings of the International Conference on Image Processing (ICIP'95), Volume 2*, 1995, 2244-2247.

[18] Lacroix, D. *Resize Images*. Acesso em 26 de Julho de 2006: http://lab.erasme.org/resize_image/index.html

[19] Ackerman, M. S. The Intellectual Challenge of CSCW: The Gap Between Social Requirements and Technical Feasibility. In *Human-Computer Interaction 15, 2&3* (2000), 179-203.