O Estado Atual dos Multiterminais

Paulo Ricardo Zanoni[1], **Tiago Vignatti**[1],
Ander Conselvan de Oliveira[1], **Fabiano Silva**[1], **Luis Carlos Erpen de Bona**[1]

[1]Departamento de Informática – Universidade Federal do Paraná
Caixa Postal 19.081 – CEP 81.531-980 – Curitiba – PR – Brasil

{paulo, vignatti, ander, fabiano, bona}@c3sl.ufpr.br

Abstract. *A multiterminal, or multiseat, is a computer that has multiple sets of input and output devices, which can be associated in order to offer independent terminals. This paper presents the graphical arquitecture of the current computers, showing that it is inadequate to the multiseat model. Existing alternatives to implement multiseat are also presented. Considering the already existing solutions, two new alternatives to multiseat implementation are showed.*

Resumo. *Um multiterminal, ou multiseat, é um computador que possui múltiplos conjuntos de dispositivos de entrada e saída que podem ser associados de forma a oferecer terminais independentes. Este artigo apresenta a arquitetura gráfica dos computadores atuais, mostrando que esta é inadequada para o modelo do multiterminal. Também são apresentadas as alternativas existentes para implementar multiterminais. Considerando os problemas das soluções já existentes, são apresentadas duas novas alternativas para implementação de multiterminais.*

1. Introdução

Um multiterminal é um computador pessoal (PC) que suporta vários usuários localmente e simultaneamente. Uma configuração típica consiste de vários conjuntos de dispositivos de entrada (ex. mouse, teclado, chaveiro USB, câmera) e de saída (ex. monitor), onde cada conjunto pertence a um usuário, o qual opera a máquina de maneira independente, i.e, cada usuário usando sua respectiva sessão [Oliveira et al. 2006].

O objetivo do modelo multiterminal é aproveitar os recursos ociosos do computador. Enquanto um usuário está lendo uma página da Internet – e portanto utilizando poucos recursos da máquina –, outro poderia estar fazendo uma atividade que envolve um uso mais intensivo dos recursos, como por exemplo jogando *games*. Outras vantagens desse modelo são o custo reduzido de instalação e manutenção, a economia de energia, a necessidade de menos pontos de rede e tomadas e menor poluição sonora. O projeto Paraná Digital [Castilho et al. 2006] é um grande caso de uso, onde 11 mil multiterminais estão sendo instalados em todas as escolas públicas estaduais do Paraná, totalizando 44 mil pontos de trabalho.

Entretanto a implementação do multiterminal é dificultada pela concepção dos computadores pessoais que foram projetados para possuírem um único usuário. Desta limitação derivam duas dificuldades: primeiro, os computadores geralmente são configurados com apenas com um dispositivo de vídeo e portanto configurações com vários desses dispositivos nem sempre são totalmente suportadas; segundo, apesar da facilidade

para conectar diversos dispositivos de entrada, como teclados e mouses em um único computador, os mecanismos para associar um conjunto de dispositivos a um determinado usuário e monitor de vídeo não são suficientemente desenvolvidos.

Atualmente existem algumas soluções para implementar o modelo do multiterminal no GNU/Linux, porém elas podem facilmente ser portadas para outros sistemas operacionais. Podemos dividir tais soluções em dois grupos: as que utilizam várias instâncias do servidor Xorg e as que utilizam servidores X aninhados. As soluções do primeiro grupo são instáveis porém apresentam o melhor desempenho. Por outro lado, as soluções do segundo grupo são estáveis mas têm um desempenho pior. Neste artigo são apresentadas duas novas maneiras de conceber multiterminais que visam suprir essas deficiências.

O restante deste artigo está organizado da seguinte forma. A Seção 2 apresenta os problemas inerentes à arquitetura dos multiterminais, bem como a primeira solução proposta. As soluções atuais para esses problemas, juntamente com suas vantagens e desvantagens são explicadas na Seção 3. Duas novas propostas de soluções são expostas na Seção 4 e a conclusão desse trabalho é apresentada na Seção 5.

2. Problemas inerentes à arquitetura gráfica

O *X Window System*, comumente chamado de X11 ou X, é um protocolo que possibilita o uso de interfaces gráficas compostas de janelas através da rede [Scheifler and Gettys 1997]. O X utiliza o modelo cliente-servidor para fazer a comunicação, onde os clientes são as aplicações – como navegador web, gerenciador de arquivos e outros – e o servidor é responsável por controlar os dispositivos de entrada e as placas gráficas (Figura 1). Existem várias implementações do servidor X [Angebranndt et al. 2004]. A implementação referência é a disponibilizada pela X.Org Foundation, denominada *Xorg* [Xorg 2008].

O canal de comunicação entre o servidor X e seus clientes é chamado de *display*. Assume-se que há apenas um usuário utilizando cada *display* e o servidor Xorg não possui suporte a múltiplos *displays* (ou *multi-display*)[1]. Dessa maneira, para montar um sistema multiterminal, a solução trivial seria executar um Xorg para cada usuário, sendo cada servidor responsável por controlar um teclado, mouse e placa de vídeo. Essa foi umas das primeiras soluções implementadas e revelou muitos problemas.

No início não era possível separar os eventos de entrada recebidos de acordo com o dispositivo que os emitiu. Logo, não era possível separar um conjunto de teclado e mouse para cada instância do Xorg. No entanto, esse problema foi superado através da criação de uma nova camada de eventos de entrada no Kernel Linux, conhecida como *evdev*, e um driver dentro do Xorg para utilizá-la. Infelizmente isso ainda não resolvia todos os problemas de iniciar várias instâncias do Xorg. Devido a problemas na interface VGA (*Video Graphics Array*) legada, quando o Xorg a utilizava para comunicar-se com sua placa de vídeo, era possível que ele acabasse mandando sua mensagem para outras placas além daquela sobre a qual ele era responsável, o que deixava o sistema em um estado inconsistente.

[1]Inicialmente o Xorg não tinha suporte a vários monitores e o termo *display* – que na lingua inglesa significa monitor – fazia mais sentido. Hoje esse termo é ainda utilizado e facilmente confunde os leitores, pois um servidor com vários monitores não é comumente chmado de *multi-display*, mas sim de *multi-head*.

A interface VGA legada define um protocolo de comunicação entre as placas de vídeo e o processador [Shanley and Anderson 1999]. Essa comunicação é feita através de endereços físicos fixos de memória. Quando há mais de uma placa de vídeo, uma mensagem enviada para um desses endereços de memória será recebida por todas as placas, a não ser que a decodificação dessa interface seja desabilitada na própria placa ou nas pontes PCI (*Peripheral Component Interconnect*) . Portanto, se não houver um processo controlando e ordenando todos os acessos feitos utilizando essa interface, multiterminais utilizando múltiplas instâncias do Xorg não serão suficientemente estáveis. Para o caso de apenas um servidor Xorg controlando várias placas de vídeo (*multi-head*), esse controle é feito pelo módulo chamado *Resource Access Control* (RAC).

Apesar disso, alguns dispositivos de vídeo mais modernos não precisam utilizar a interface VGA legada, portanto é possível fazer multiterminais da maneira descrita acima (essas soluções foram apresentas em [Oliveira et al. 2006]). Entretanto, como essa solução só funciona com certas placas e também somente com drivers específicos – em alguns casos apenas os drivers proprietários –, ela não é satisfatória.

3. Soluções existentes

As primeiras soluções estáveis de multiterminal – que funcionam com todos dispositivos gráficos suportados pelo Xorg – utilizam servidores X aninhados. Um servidor X aninhado utiliza como dispositivos de entrada e saída um outro servidor X. Dessa maneira, um servidor X aninhado delega a tarefa de acesso ao hardware para o servidor sobre o qual roda, fazendo requisições através do protocolo X11, ilustrado na Figura 2.

3.1. Xnest

A primeira solução multiterminal que utiliza servidores X aninhados foi concebida pelo Centro de Computação Científica e Software Livre (C3SL) no final de 2005 [Oliveira et al. 2006]. No servidor *Xnest*, foi implementado um novo driver de entrada, o qual recebe os eventos de entrada diretamente da interface *evdev* do Kernel Linux, ao invés de recebê-los do servidor base. Além disso, um cursor de mouse próprio foi também implementado (o Xnest original utiliza o cursor do servidor base).

Entretanto essa solução apresenta alguns problemas: (i) a falta de desenvolvimento do Xnest pela comunidade implica em falta de manutenção do mesmo; (ii) a falta de suporte à extensões muito utilizadas nos ambientes desktop, como Render, Xv (para tocar vídeo com melhor qualidade) e GLX (quer permite utilizar melhor o hardware de vídeo para operações 3D) e (iii) a sobrecarga no processamento das requisições gráficas pelo servidor base, pois o Xnest é apenas um *proxy*.

3.2. Xephyr

Diferente do Xnest, o Xephyr utiliza a janela do servidor base como se fosse um *framebuffer* próprio, através de memória compartilhada quando possível. Isso permite que as requisições sejam tratadas no próprio Xephyr, diminuindo a sobrecarga de executar servidores aninhados.

O Xephyr foi construído sobre a arquitetura Kdrive [Xephyr 2008], que facilita a implementação de extensões X11. Por essa razão, ele implementa as extensões Render e

Shm, permitindo que desktops modernos tenham um melhor desempenho. Recentemente ele foi incrementado com as extensões Xv e GLX.

A modificação necessária no Xephyr para utilizá-lo nos multiterminais foi a criação de um novo driver de entrada específico da arquitetura Kdrive, o *evdev*. Atualmente, essa é a solução mais empregada nos multiterminais existentes.

3.3. Xglx

A terceira solução multiterminal com servidores aninhados utiliza o Xglx, que é baseado na arquitetura Xgl [Xgl 2008]. Essa arquitetura ganhou notoriedade no ano de 2006 pelo seu desempenho ao executar aplicações OpenGL utilizando o dispositivo de vídeo, o que permite uma grande variedade de efeitos visuais interessantes.

O Xglx é o *backend* do Xgl, que é executado de maneira aninhada sobre um servidor X normal e aproveita-se da extensão GLX do servidor base para disponibilizar o conteúdo gráfico. A modificação necessária no Xglx para que fosse utilizado como multiterminal foi semelhante à do Xephyr.

A grande desvantagem do Xglx é a falta de suporte da comunidade, pois não existe mais interesse em mantê-lo devido ao AIGLX, um outro método para disponibilizar efeitos visuais 3D que utiliza o servidor Xorg ao invés do Xgl. Além disso, a placa de vídeo precisa possuir um hardware gráfico 3D e uma quantidade razoável de memória, o que atualmente nem todos dispositivos de baixo custo possuem.

3.4. Xat

No início de 2007, o C3SL iniciou o desenvolvimento do *X Address Translation* (Xat) [Xat 2007]. O Xat é uma aplicação *proxy* que atua como um intermediário entre clientes X em um servidor X multi-head. Para cada placa de vídeo existente no servidor X, o Xat abre um *display*, dando a impressão de que há múltiplos servidores X na máquina. Conforme demonstrado na Figura 3, toda a comunicação entre os clientes e o servidor é intermediada pelo Xat, de modo que, do ponto de vista do servidor X, apenas o Xat é cliente, e do ponto de vista das aplicações, o Xat é o servidor X.

Para o correto funcionamento dos vários *displays*, o Xat altera as requisições enviadas pelos clientes e os eventos e respostas recebidos do servidor X. Dessa maneira, sua sobrecarga em relação aos outros servidores aninhados tende a ser menor, uma vez que ele não é uma implementação completa de um servidor X. Além disso, o Xat permite que os clientes utilizem as extensões implementadas no servidor X diretamente. No entanto, algumas extensões podem não funcionar corretamente sem alterações em suas requisições, exigindo a implementação de código específico dentro do Xat.

Outro problema é que ele utiliza o *Multi-Pointer X server* (MPX) [MPX 2007] para a sua camada de entrada, que é uma modificação do servidor Xorg, que está ainda instável e será integrada ao mesmo futuramente. Atualmente ainda não é possível utilizar a solução do Xat, pois ele está inacabado e o seu desenvolvimento foi congelado.

4. As novas soluções

Conforme visto nas seções anteriores, ao utilizar múltiplas instâncias do servidor Xorg, problemas originados pela interface VGA legada podem levar a máquina a um estado inconsistente. Por outro lado, ao executar apenas uma instância do Xorg que controla todas

Figura 1. Uso normal do servidor X.

Figura 2. Servidores X aninhados.

Figura 3. Multiterminal com Xat.

as placas de vídeo e usar vários servidores aninhados, temos que a falta de extensões X e a sobrecarga dos servidores aninhados deixam a máquina com um desempenho pouco satisfatório, apesar de estável. Esses problemas motivaram duas novas maneiras de criar multiterminais. A primeira, apresentada na Seção 4.1, é apenas descrita porém não implementada. Já a segunda, na Seção 4.2, está sendo desenvolvida e os resultados preliminares são promissores.

4.1. X com múltiplos displays

Conforme explicado anteriormente, as aplicações e o Xorg foram criados assumindo que há apenas um usuário utilizando cada display, e se houver mais de um usuário por display surgem inúmeros desafios, inclusive problemas de segurança. Portanto, para que vários usuários possam utilizar o mesmo servidor Xorg de maneira segura, este deverá prover múltiplos displays, um para cada usuário.

Dessa maneira, a sobrecarga causada pelos servidores aninhados seria eliminada, pois não haveria nenhuma aplicação entre o servidor X, que comunica-se diretamente com o hardware, e os clientes X. Além disso, não haveria a necessidade de reimplementação de extensões nos servidores aninhados.

Uma possível impelementação seria incorporar o Xat ao Xorg. Com isso, as tarefas que o Xat realiza como proxy seriam realizadas dentro do servidor X, diminuindo a latência e proporcionando um Xorg multi-display. Porém, essa solução ainda não foi implementada, já que os esforços da comunidade estão concentrados na solução descrita a seguir.

4.2. O Árbitro VGA

O Árbitro VGA [Arbiter VGA] é a entidade responsável por controlar todos os acessos feitos às placas de vídeo que utilizam a interface VGA legada. Todas as aplicações que utilizam essa interface devem antes requisitar o acesso a ele, que garantirá que apenas a placa de vídeo desejada decodifique o acesso. É a implementação de um semáforo. Com ele, é possível executar múltiplas instâncias do Xorg, o que possibilita a criação de multiterminais com um servidor para cada usuário, conforme descrito na Seção 2.

Em sua implementação atual, o Árbitro VGA é um módulo do Kernel Linux, que pode ser acessado tanto dentro do próprio Kernel, através de um conjunto de funções, quanto de uma biblioteca em espaço de usuário, que utiliza um *device node* para realizar a comunicação. Essas interfaces ainda estão em desenvolvimento e não estão estáveis o suficiente para colocar um multiterminal em produção. Há também uma implementação do servidor Xorg que utiliza o árbitro, porém ela é apenas uma prova de conceito, por ser

um simples invólucro nas chamadas de função dos drivers de vídeo. A implementação definitiva deverá ser feita dentro dos próprios drivers, por questões de desempenho.

Testes iniciais mostraram que apesar de simples e não otimizada, a implementação atual apresenta uma perda de desempenho insignificante, além, é claro, de conseguir realizar com sucesso o seu propósito.

5. Considerações Finais

Este artigo mostrou os problemas inerentes ao modelo multiterminal de computação, bem como as soluções já existentes, suas vantagens, desvantagens e apresentou duas novas soluções. O árbitro VGA é uma solução promissora e permitirá um modelo de computação de baixo custo, que pode ser combinado com outras soluções – como centralização de discos, centralização de processamento, dentre outras – com o intuito de produzir um modelo melhor ainda e mais acessível.

Apesar de todos os problemas relacionados às soluções atuais expostas neste trabalho, os multiterminais são cada vez mais utilizados, principalmente em grandes parques computacionais, devido ao seu baixo custo de implantação e manutenção. O grande entrave dos multiterminais continua sendo a tentativa contínua de deixar esse modelo transparente ao usuário, de modo que ele pense que está usando um computador pessoal comum. Várias aplicações ainda deverão ser portadas para se beneficiar desse modelo.

Referências

Angebranndt, S., Drewry, R., Karlton, P., Newman, T., Scheifler, B., Packard, K., Wiggins, D., and Gettys, J. (2004). Definition of the Porting Layer for X v11 Sample Server. *X Consortium Inc and X.Org Foundation*.

Arbiter VGA. http://www.x.org/wiki/VgaArbiter. Acesso em: Março, 2008.

Castilho, M. A., Sunye, M. S., Weingaerter, D., Bona, L. C. E., Silva, F., Direne, A. I., Garcia, L. S., and Guedes, A. L. P. (2006). *Making government policies for education possible by means of Open Source technology: a successful case*, volume 1. Hershey - USA: Idea Group Publishing, 1st edition.

MPX (2007). http://wearables.unisa.edu.au/mpx/. Acesso em: Março, 2008.

Oliveira, A. C., Vignatti, T., Weigaertner, D., Silva, F., Castilho, M., and Sunye, M. (2006). Um modelo de computação multiusuário baseado em computadores pessoais. *VII Workshop sobre Software Livre*, pages 135–140.

Scheifler, R. W. and Gettys, J. (1997). *X Window System: core and extension protocols*. Digital Equipment Corp., Acton, MA, USA.

Shanley, T. and Anderson, D. (1999). *PCI System Architecture*. Addison-Wesley, 4th edition.

Xat (2007). http://www.c3sl.ufpr.br/multiseat/xat/. Acesso em: Março, 2008.

Xephyr (2008). http://www.freedesktop.org/wiki/Software/Xephyr. Acesso em: Março, 2008.

Xgl (2008). http://www.freedesktop.org/wiki/Software/Xgl. Acesso em: Março, 2008.

Xorg (2008). http://www.x.org. Acesso em: Março, 2008.