O que é um sistema operacional?

A maioria dos computadores tem dois modos de operação: modo núcleo e modo usuário. O sistema operacional, a peça mais fundamental de software, opera em modo núcleo (também chamado modo supervisor). Nesse modo ele tem acesso completo a todo o hardware e pode executar qualquer instrução que a máquina for capaz de executar. O resto do software opera em modo usuário, no qual apenas um subconjunto das instruções da máquina está disponível. Em particular, aquelas instruções que afetam o controle da máquina ou realizam E/S (Entrada/Saída) são proibidas para programas de modo usuário.

Os sistemas operacionais são enormes, complexos e têm vida longa. O código-fonte do coração de um sistema operacional como Linux ou Windows tem cerca de cinco milhões de linhas

Os sistemas operacionais realizam duas funções essencialmente não relacionadas: fornecer a programadores de aplicativos (e programas aplicativos, claro) um conjunto de recursos abstratos limpo em vez de recursos confusos de hardware, e gerenciar esses recursos de hardware.

Um software, chamado driver de disco, lida com o hardware e fornece uma interface para ler e escrever blocos de dados, sem entrar nos detalhes. Sistemas operacionais contêm muitos drivers para controlar dispositivos de E/S. Todos os sistemas operacionais fornecem mais um nível de abstração para se utilizarem discos: arquivos. Usando essa abstração, os programas podem criar, escrever e ler arquivos, sem ter de lidar com os detalhes complexos de como o hardware realmente funciona.

É muito mais fácil lidar com fotos, e-mails, músicas e páginas da web do que com detalhes de discos SATA (ou outros). A função dos sistemas operacionais é criar boas abstrações e então implementar e gerenciar os objetos abstratos criados desse modo.

O gerenciamento de recursos inclui a multiplexação (compartilhamento) de recursos de duas maneiras diferentes: no tempo e no espaço.

• Quando um recurso é multiplexado no tempo, diferentes programas ou usuários se revezam usando-o.

• O outro tipo é a multiplexação de espaço. Em vez de os clientes se revezarem, cada um tem direito a uma parte do recurso.

História dos sistemas operacionais

O primeiro computador verdadeiramente digital foi projetado pelo matemático inglês Charles Babbage (1792– 1871). Mas Babbage nunca conseguiu colocá-lo para funcionar para valer porque a máquina era puramente mecânica. A solução para esse problema foi dada pela

jovem Ada Lovelace, a primeira programadora do mundo. A linguagem de programação Ada® é uma homenagem a ela.

• *A primeira geração (1945-1955): válvulas*: No início, um único grupo de pessoas (normalmente engenheiros) projetava, construía, programava, operava e mantinha cada máquina. Toda a programação era feita em código de máquina absoluto, ou, pior ainda, ligando circuitos elétricos através da conexão de milhares de cabos a painéis de ligações para controlar as funções básicas da máquina. Linguagens de programação eram desconhecidas (mesmo a linguagem de montagem era desconhecida). Ninguém tinha ouvido falar ainda de sistemas operacionais. O modo usual de operação consistia na reserva pelo programador de um bloco de tempo na ficha de registro na parede, então ele descer até a sala de máquinas, inserir seu painel de programação no computador e passar as horas seguintes torcendo para que nenhuma das cerca de 20.000 válvulas queimasse durante a operação. Virtualmente todos os problemas eram cálculos numéricos e matemáticos diretos e simples, como determinar tabelas de senos, cossenos e logaritmos, ou calcular trajetórias de artilharia. No início da década de 1950, a rotina havia melhorado de certa maneira com a introdução dos cartões perfurados. Era possível agora escrever programas em cartões e lê-los em vez de se usarem painéis de programação; de resto, o procedimento era o mesmo.

• *A segunda geração (1955-1965): transistores e sistemas em lote (batch):* Para executar uma tarefa [isto é, um programa ou conjunto de programas], um programador primeiro escrevia o programa no papel [em FORTRAN ou em linguagem de montagem (assembly)], então o perfurava nos cartões. Ele levava então o maço de cartões até a sala de entradas e o passava a um dos operadores e ia tomar um café até que a saída estivesse pronta. Quando o computador terminava qualquer tarefa que ele estivesse executando no momento, um operador ia até a impressora, pegava a sua saída e a levava até a sala de saídas a fim de que o programador pudesse buscá-la mais tarde. Então ele pegava um dos maços de cartões que haviam sido levados da sala de entradas e o colocava para a leitura. Se o compilador FORTRAN fosse necessário, o operador teria de tirá-lo de um porta-arquivos e fazer a leitura. Muito tempo do computador era desperdiçado enquanto os operadores caminhavam em torno da sala de máquinas. Dado o alto custo do equipamento, não causa surpresa que as pessoas logo procuraram maneiras de reduzir o tempo desperdiçado. A solução geralmente adotada era o sistema em lote (batch). A ideia por trás disso era reunir um lote de tarefas na sala de entradas e então passá-lo para uma fita magnética usando um computador pequeno e (relativamente) barato, como um IBM 1401, que era muito bom na leitura de cartões, cópia de fitas e impressão de saídas, mas ruim em cálculos numéricos.

• *A terceira geração (1965-1980): CIs e multiprogramação:* O IBM 360 foi a primeira linha importante de computadores a usar CIs (circuitos integrados) de pequena escala, proporcionando desse modo uma vantagem significativa na relação preço/desempenho sobre as máquinas de segunda geração, que foram construídas sobre transistores individuais. O forte da ideia da “família única” foi ao mesmo tempo seu maior ponto fraco. Ele tinha de funcionar bem em sistemas com poucos periféricos e naqueles com muitos periféricos, além de ambientes comerciais e ambientes científicos, o que era impossível.

A multiprogramação: A solução encontrada foi dividir a memória em várias partes, com uma tarefa diferente em cada partição, como mostrado na Figura 1.5. Enquanto uma tarefa ficava esperando pelo término da E/S, outra podia usar a CPU. Se um número suficiente de tarefas pudesse ser armazenado na memória principal ao mesmo tempo, a CPU podia se manter ocupada quase 100% do tempo.

Embora sistemas operacionais de terceira geração fossem bastante adequados para grandes cálculos científicos e operações maciças de processamento de dados comerciais, eles ainda eram basicamente sistemas em lote. Muitos programadores sentiam saudades dos tempos de computadores de primeira geração quando eles tinham a máquina só para si por algumas horas e assim podiam corrigir eventuais erros em seus programas rapidamente. Com sistemas de terceira geração, o tempo entre submeter uma tarefa e receber de volta a saída era muitas vezes de várias horas, então uma única vírgula colocada fora do lugar podia provocar a falha de uma compilação, e o desperdício de metade do dia do programador. Os programadores não gostavam muito disso. Esse desejo por um tempo de resposta rápido abriu o caminho para o timesharing (compartilhamento de tempo), uma variante da multiprogramação, na qual cada usuário tem um terminal on-line. Em um sistema de timesharing, se 20 usuários estão conectados e 17 deles estão pensando, falando ou tomando café, a CPU pode ser alocada por sua vez para as três tarefas que demandam serviço.

O MULTICS foi um sucesso relativo. Ele foi projetado para suportar centenas de usuários em uma máquina apenas um pouco mais poderosa do que um PC baseado no 386 da Intel, embora ele tivesse muito mais capacidade de E/S. Apesar da falta de sucesso comercial, o MULTICS teve uma influência enorme em sistemas operacionais subsequentes (especialmente UNIX e seus derivativos, FreeBSD, Linux, iOS e Android).

• *A quarta geração (1980-presente): computadores pessoais:* Com o desenvolvimento dos circuitos integrados em larga escala (Large Scale Integration — LSI) — que são chips contendo milhares de transistores em um centímetro quadrado de silicone —, surgiu a era do computador moderno.

• *A quinta geração (1990-presente): computadores móveis:* O primeiro telefone verdadeiramente móvel foi criado na década de 1970 e, pesando cerca de um quilo, era positivamente um peso-pena. Ele ficou conhecido carinhosamente como “o tijolo”. Logo todos queriam um. Hoje, a penetração do telefone móvel está próxima de 90% da população global.

Revisão sobre hardware de computadores

Processador

O “cérebro” do computador é a CPU. O ciclo básico de toda CPU é buscar a primeira instrução da memória, decodificá-la para determinar o seu tipo e operandos, executá-la, e

então buscar, decodificar e executar as instruções subsequentes. O ciclo é repetido até o programa terminar.

Para melhorar o desempenho, os projetistas de CPU há muito tempo abandonaram o modelo simples de buscar, decodificar e executar uma instrução de cada vez. Muitas CPUs modernas têm recursos para executar mais de uma instrução ao mesmo tempo. Por exemplo, uma CPU pode ter unidades de busca, decodificação e execução separadas, assim enquanto ela está executando a instrução n, poderia também estar decodificando a instrução n + 1 e buscando a instrução n + 2. Uma organização com essas características é chamada de pipeline.

Chips multithread e multinúcleo: o Pentium 4 da Intel introduziu a propriedade chamada multithreading ou hyperthreading (o nome da Intel para ela), ao processador x86 e vários outros chips de CPU também o têm.

Para uma primeira aproximação, o que ela faz é permitir que a CPU mantenha o estado de dois threads diferentes e então faça o chaveamento entre um e outro em uma escala de tempo de nanossegundos, vale ressaltar que o multithreading não proporciona paralelismo real.

Uma GPU é um processador com, literalmente, milhares de núcleos minúsculos. Eles são muito bons para realizar muitos pequenos cálculos feitos em paralelo, como reproduzir polígonos em aplicações gráficas. Não são tão bons em tarefas em série. Eles também são difíceis de programar. Embora GPUs possam ser úteis para sistemas operacionais (por exemplo, codificação ou processamento de tráfego de rede), não é provável que grande parte do sistema operacional em si vá ser executada nas GPUs.

Memória

Memória: é o segundo principal componente em qualquer computador, o qual deve ser rápido ao extremo (mais rápida do que executar uma instrução, de maneira que a CPU não seja atrasada pela memória).

A camada superior consiste em registradores internos à CPU. Eles são feitos do mesmo material que a CPU e são, desse modo, tão rápidos quanto ela.Em seguida, vem a memória cache, que é controlada principalmente pelo hardware. A memória principal é dividida em linhas de cache, tipicamente 64 bytes, com endereços 0 a 63 na linha de cache 0, 64 a 127 na linha de cache 1 e assim por diante.

A memória principal vem a seguir na hierarquia da Figura 1.9. Trata-se da locomotiva do sistema de memória. A memória principal é normalmente chamada de RAM. Além da memória principal, muitos computadores têm uma pequena memória de acesso aleatório não volátil. Diferentemente da RAM, a memória não volátil não perde o seu conteúdo quando a energia é desligada. A ROM (Read Only Memory — memória somente de leitura) é programada na fábrica e não pode ser modificada depois.

A EEPROM (Electrically Erasable PROM — ROM eletricamente apagável) e a memória flash também são não voláteis, mas, diferentemente da ROM, podem ser apagadas e reescritas. No entanto, escrevê- -las leva muito mais tempo do que escrever em RAM, então elas são usadas da mesma maneira que a ROM, apenas com a característica adicional de que é possível agora corrigir erros nos programas que elas armazenam mediante sua regravação.

A memória flash também é bastante usada como um meio de armazenamento em dispositivos eletrônicos portáteis. Ela serve como um filme em câmeras digitais e como disco em reprodutores de música portáteis, apenas como exemplo. A memória flash é intermediária em velocidade entre a RAM e o disco. Também, diferentemente da memória de disco, ela se desgasta quando apagada muitas vezes.

Outro tipo ainda de memória é a CMOS, que é volátil. Muitos computadores usam a memória CMOS para armazenar a hora e a data atualizadas. A memória CMOS e o circuito de relógio que incrementa o tempo registrado nela são alimentados por uma bateria pequena, então a hora é atualizada corretamente, mesmo quando o computador estiver desligado.

Discos

Discos: um disco consiste em um ou mais pratos metálicos que rodam a 5.400, 7.200, 10.800 RPM, ou mais. Um braço mecânico move-se sobre esses pratos a partir da lateral, como o braço de toca-discos de um velho fonógrafo de 33 RPM para tocar discos de vinil.

SSDs (Solid State Disks — discos em estado sólido). SSDs não têm partes móveis, não contêm placas na forma de discos e armazenam dados na memória (flash). A única maneira pela qual lembram discos é que eles também armazenam uma quantidade grande de dados que não é perdida quando a energia é desligada.

Muitos computadores dão suporte a um esquema conhecido como memória virtual. Esse esquema torna possível executar programas maiores que a memória física colocando-os no disco e usando a memória principal como um tipo de cache para as partes mais intensivamente executadas. Esse esquema exige o remapeamento dos endereços de memória rapidamente para converter o endereço que o programa gerou para o endereço físico em RAM onde a palavra está localizada. Esse mapeamento é feito por uma parte da CPU chamada MMU. A presença da cache e da MMU pode ter um impacto importante sobre o desempenho.

Dispositivos de E/S

Dispositivos de E/S consistem em geral em duas partes: um controlador e o dispositivo em si. O controlador é um chip ou um conjunto de chips que controla fisicamente o dispositivo.

Ele aceita comandos do sistema operacional, por exemplo, para ler dados do dispositivo, e os executa.

Os dispositivos possuem interfaces relativamente simples, tanto porque eles não podem fazer muito, como para padronizá-los. A padronização é necessária para que qualquer controlador de disco SATA possa controlar qualquer disco SATA, por exemplo.

O software que conversa com um controlador, dando a ele comandos e aceitando respostas, é chamado de driver de dispositivo.

Dispositivos de E/S: é comum utilizar um controlador do dispositivo associado ao controlador de interrupções.

A entrada e a saída podem ser realizadas de três maneiras diferentes. No método mais simples, um programa de usuário emite uma chamada de sistema, que o núcleo traduz em uma chamada de rotina para o driver apropriado. O driver então inicia a E/S e aguarda usando um laço curto, inquirindo continuamente o dispositivo para ver se ele terminou a operação (em geral há algum bit que indica que o dispositivo ainda está ocupado). Quando a operação de E/S termina, o driver coloca os dados (se algum) onde eles são necessários e retorna. O sistema operacional então retorna o controle a quem o chamou. Esse método é chamado de espera ocupada e tem a desvantagem de manter a CPU ocupada interrogando o dispositivo até o término da operação de E/S. No segundo método, o driver inicia o dispositivo e pede a ele que o interrompa quando tiver terminado. Nesse ponto, o driver retorna. O sistema operacional bloqueia então o programa que o chamou, se necessário, e procura por mais trabalho para fazer. Quando o controlador detecta o fim da transferência, ele gera uma interrupção para sinalizar o término.

Barramentos

Barramentos: à medida que os processadores e as memórias foram ficando mais rápidos, a capacidade de um único barramento de lidar com todo o tráfego foi exigida até o limite. Barramentos adicionais foram acrescentados, tanto para dispositivos de E/S mais rápidos quanto para o tráfego CPU para memória.

Este sistema tem muitos barramentos (por exemplo, cache, memória, PCIe, PCI, USB, SATA e DMI), cada um com uma taxa de transferência e função diferentes.

Uma arquitetura de barramento compartilhado significa que múltiplos dispositivos usam os mesmos fios para transferir dados. Assim, quando múltiplos dispositivos têm dados para enviar, você precisa de um árbitro para determinar quem pode utilizar o barramento. Em comparação, o PCIe faz uso de conexões dedicadas de ponto a ponto. Uma arquitetura de barramento paralela como usada no PCI tradicional significa que você pode enviar uma palavra de dados através de múltiplos fios. Por exemplo, em barramentos PCI regulares, um único número de 32 bits é enviado através de 32 fios paralelos. Em comparação com isso, o PCIe usa uma arquitetura de barramento serial e envia todos os bits em uma mensagem através de uma única conexão, chamada faixa, de maneira muito semelhante a um pacote de rede. Isso é muito mais simples, pois você não tem de assegurar que todos os 32 bits cheguem ao destino exatamente ao mesmo tempo. O paralelismo ainda é usado, pois você pode ter múltiplas faixas em paralelo. Por exemplo, podemos usar 32 faixas para carregar 32 mensagens em paralelo.

O USB (Universal Serial Bus — barramento serial universal) foi inventado para conectar todos os dispositivos de E/S lentos, como o teclado e o mouse, ao computador.

O barramento SCSI (Small Computer System Interface — interface pequena de sistema computacional) é um barramento de alto desempenho voltado para discos rápidos, digitalizadores de imagens e outros dispositivos que precisam de uma considerável largura de banda.

O plug and play faz o sistema coletar automaticamente informações sobre os dispositivos de E/S, atribuir centralmente níveis de interrupção e endereços desses dispositivos e, então, informar a cada placa quais são os seus números. Esse trabalho está relacionado de perto à inicialização do computador, então vamos examinar essa questão. Ela não é completamente trivial.

O zoológico dos sistemas operacionais

• *Sistemas operacionais de computadores de grande porte*: No topo estão os sistemas operacionais para computadores de grande porte (mainframes), aquelas máquinas do tamanho de uma sala ainda encontradas nos centros de processamento de dados de grandes corporações. Esses computadores diferem dos computadores pessoais em termos de sua capacidade de E/S. Os sistemas operacionais para computadores de grande porte são intensamente orientados para o processamento de muitas tarefas ao mesmo tempo, a maioria delas exigindo quantidades prodigiosas de E/S. Eles em geral oferecem três tipos de serviços: em lote (batch), processamento de transações e tempo compartilhado (timesharing).

• *Sistemas operacionais de servidores*: Um nível abaixo estão os sistemas operacionais de servidores. Eles são executados em servidores que são computadores pessoais muito grandes, em estações de trabalho ou mesmo computadores de grande porte. Eles servem a múltiplos usuários ao mesmo tempo por meio de uma rede e permitem que os usuários compartilhem recursos de hardware e software. Servidores podem fornecer serviços de impressão, de arquivo ou de web.

• *Sistemas operacionais de multiprocessadores:* Uma maneira cada vez mais comum de se obter potência computacional para valer é conectar múltiplas CPUs a um único sistema.

• *Sistemas operacionais de computadores pessoais:* Todos os computadores modernos dão suporte à multiprogramação, muitas vezes com dezenas de programas iniciados no momento da inicialização do sistema. Seu trabalho é proporcionar um bom apoio para um único usuário.

• *Sistemas operacionais de computadores portáteis*: Seguindo com sistemas cada vez menores, chegamos aos tablets, smartphones e outros computadores portáteis. Um computador portátil, originalmente conhecido como um PDA (Personal Digital Assistant — assistente pessoal digital), é um computador pequeno que pode ser seguro na mão durante a operação.

• *Sistemas operacionais embarcados:* Sistemas embarcados são executados em computadores que controlam dispositivos que não costumam ser vistos como computadores e que não aceitam softwares instalados pelo usuário. Exemplos típicos são os fornos de micro-ondas, os aparelhos de televisão, os carros, os aparelhos de DVD, os telefones tradicionais e os MP3 players. A principal propriedade que distingue sistemas embarcados dos portáteis é a certeza de que nenhum software não confiável vá ser executado nele um dia. Você não consegue baixar novos aplicativos para o seu forno de micro-ondas – todo o software está na memória ROM. Isso significa que não há necessidade para proteção entre os aplicativos, levando a simplificações no design.

• *Sistemas operacionais de nós sensores (senso r-node)*: Redes de nós sensores minúsculos estão sendo empregadas para uma série de finalidades. Esses nós são computadores minúsculos que se comunicam entre si e com uma estação-base usando comunicação sem fio. Redes de sensores são usadas para proteger os perímetros de prédios, guardar fronteiras nacionais, detectar incêndios em florestas, medir a temperatura e a precipitação para a previsão de tempo, colher informações sobre a movimentação de inimigos nos campos de batalha e muito mais.

• *Sistemas operacionais de tempo real*: Esses sistemas são caracterizados por ter o tempo como um parâmetro-chave. Por exemplo, em sistemas de controle de processo industrial, computadores em tempo real têm de coletar dados a respeito do processo de produção e usá-los para controlar máquinas na fábrica. Muitas vezes há prazos rígidos a serem cumpridos. Por exemplo, se um carro está seguindo pela linha de montagem, determinadas ações têm de ocorrer em dados instantes. Se, por exemplo, um robô soldador fizer as soldas cedo demais ou tarde demais, o carro será arruinado. Se a ação tem de ocorrer absolutamente em um determinado momento (ou dentro de uma dada faixa de tempo), temos um sistema de tempo real crítico. Muitos desses sistemas são encontrados no controle de processos industriais, aviônica, militar e áreas de aplicação semelhantes. Esses sistemas têm de fornecer garantias absolutas de que uma determinada ação ocorrerá em um determinado momento.

Um sistema de tempo real não crítico é aquele em que perder um prazo ocasional, embora não desejável, é aceitável e não causa danos permanentes. Sistemas de multimídia ou

áudio digital caem nesta categoria. Smartphones também são sistemas de tempo real não críticos.

• *Sistemas operacionais de cartões inteligentes (smartcard):* Os menores sistemas operacionais são executados em cartões inteligentes, que são dispositivos do tamanho de cartões de crédito contendo um chip de CPU. Possuem severas restrições de memória e processamento de energia. Alguns obtêm energia por contatos no leitor no qual estão inseridos, mas cartões inteligentes sem contato obtêm energia por indução, o que limita muito o que eles podem fazer. Alguns deles conseguem realizar somente uma função, como pagamentos eletrônicos, mas outros podem realizar múltiplas funções. Muitas vezes são sistemas proprietários.

Conceitos de sistemas operacionais

Processos

Programa em execução

SO aloca recursos para cada processo:
espaço de endereçamento, registradores (contador de programa), lista de arquivos abertos, alarmes pendentes, lista de processos relacionados, dentre outras informações.

Hierarquia: O processo A criou dois processos filhos, B e C. O processo B criou três processos filhos, D, E e F.

Memória

Os primeiros computadores de grande porte tinham uma memória muito limitada. Um IBM 7090 ou um 7094 completamente carregados, que eram os melhores computadores do final de 1959 até 1964, tinha apenas um pouco mais de 128 KB de memória.

Todo computador tem alguma memória principal que ele usa para armazenar programas em execução. Em um sistema operacional muito simples, apenas um programa de cada vez está na memória.

Arquivos

É o “local” onde as informações de “longo prazo”! São armazenadas.

Os sistemas de arquivos definem como será a representação lógica dos dispositivos de armazenamento.

Diretório é a forma que o SO utiliza para agrupar arquivos. Diretório raiz e diretório de trabalho.

Antes que um arquivo possa ser lido ou escrito, ele precisa ser aberto, momento em que as permissões são conferidas. Se o acesso for permitido, o sistema retorna um pequeno valor inteiro, chamado descritor de arquivo, para usá-lo em operações subsequentes. Se o acesso for proibido, um código de erro é retornado.

Outro conceito importante em UNIX é o arquivo especial. Arquivos especiais permitem que dispositivos de E/S se pareçam com arquivos. Dessa maneira, eles podem ser lidos e escritos com as mesmas chamadas de sistema que são usadas para ler e escrever arquivos. Existem dois tipos especiais: arquivos especiais de bloco e arquivos especiais de caracteres. Arquivos especiais de bloco são usados para modelar dispositivos que consistem em uma coleção de blocos aleatoriamente endereçáveis, como discos. Ao abrir um arquivo especial de bloco e ler, digamos, bloco 4, um programa pode acessar diretamente o quarto bloco no dispositivo, sem levar em consideração a estrutura do sistema de arquivo contido nele. De modo similar, arquivos especiais de caracteres são usados para modelar impressoras, modems e outros dispositivos que aceitam ou enviam um fluxo de caracteres. Um pipe é uma espécie de pseudoarquivo que pode ser usado para conectar dois processos.

Entrada/Saída

Existem muitos tipos de dispositivos de entrada e de saída, incluindo teclados, monitores, impressoras e assim por diante. Cabe ao sistema operacional gerenciá-los. Em consequência, todo sistema operacional tem um subsistema de E/S para gerenciar os dispositivos de E/S. Alguns softwares de E/S são independentes do dispositivo, isto é, aplicam-se igualmente bem a muitos ou a todos dispositivos de E/S.

Hardware de proteção

Os primeiros computadores de grande porte inicialmente não tinham hardware de proteção e nenhum suporte para multiprogramação, então sistemas operacionais simples eram executados neles. Esses sistemas lidavam com apenas um programa carregado manualmente por vez. Mais tarde, eles adquiriram o suporte de hardware e sistema operacional para lidar com múltiplos programas ao mesmo tempo, e então capacidades de compartilhamento de tempo completas.

O interpretador de comandos (shell)

Interface dos sistemas operacionais baseados em texto. O shell embora não faça parte do sistema operacional, ele faz um uso intensivo de muitos aspectos do sistema operacional. A maioria dos computadores pessoais usa hoje uma interface gráfica GUI. Na realidade, a GUI é apenas um programa sendo executado em cima do sistema operacional, como um shell.

Memórias grandes

Os primeiros computadores de grande porte tinham uma memória limitada. Um IBM 7090 ou um 7094 completamente carregados, que eram os melhores computadores do final de 1959 até 1964, tinha apenas um pouco mais de 128 KB de memória. Em sua maior parte, eram programados em linguagem de montagem e seu sistema operacional era escrito nessa linguagem para poupar a preciosa memória.

Hardware de proteção

Os primeiros computadores de grande porte inicialmente não tinham hardware de proteção e nenhum suporte para multiprogramação, então sistemas operacionais simples eram executados neles. Esses sistemas lidavam com apenas um programa carregado manualmente por vez. Mais tarde, eles adquiriram o suporte de hardware e sistema operacional para lidar com múltiplos programas ao mesmo tempo, e então capacidades de compartilhamento de tempo completas.

Discos

Os primeiros computadores de grande porte eram em grande parte baseados em fitas magnéticas. Eles liam um programa a partir de uma fita, compilavam-no e escreviam os resultados de volta para outra fita. Não havia discos e nenhum conceito de um sistema de arquivos. Isso começou a mudar quando a IBM introduziu o primeiro disco rígido — o RAMAC (RAndoM ACcess) em 1956.

Memória virtual

A memória virtual proporciona a capacidade de executar programas maiores do que a memória física da máquina, rapidamente movendo pedaços entre a memória RAM e o disco. Ela passou por um desenvolvimento similar, primeiro aparecendo nos computadores de grande porte, então passando para os minis e os micros.

Chamadas de sistema

• *Chamadas de sistema para gerenciamento de processos:* A chamada fork é um bom ponto para se começar a discussão. A chamada fork é a única maneira para se criar um processo novo em POSIX. Ela cria uma cópia exata do processo original, incluindo todos os descritores de arquivos, registradores — tudo. Após a fork, o processo original e a cópia (o processo pai e o processo filho) seguem seus próprios caminhos separados.

• *Chamadas de sistema para gerenciamento de arquivos:* Para ler ou escrever um arquivo, é preciso primeiro abri-lo. Essa chamada especifica o nome do arquivo a ser aberto, seja como um nome de caminho absoluto ou relativo ao diretório de trabalho, assim como um

código de O_RDONLY, O_WRONLY, ou O_RDWR, significando aberto para leitura, escrita ou ambos. Para criar um novo arquivo, o parâmetro O_CREAT é usado.

O descritor de arquivos retornado pode então ser usado para leitura ou escrita. Em seguida, o arquivo pode ser fechado por close, que torna o descritor disponível para ser reutilizado em um open subsequente.

• Chamadas de sistema para gerenciamento de diretórios: As primeiras duas chamadas, mkdir e rmdir, criam e removem diretórios vazios, respectivamente. A próxima chamada é link. Sua finalidade é permitir que o mesmo arquivo apareça sob dois ou mais nomes, muitas vezes em diretórios diferentes. Um uso típico é permitir que vários membros da mesma equipe de programação compartilhem um arquivo comum, com cada um deles tendo o arquivo aparecendo no seu próprio diretório, possivelmente sob nomes diferentes. Compartilhar um arquivo não é o mesmo que dar a cada membro da equipe uma cópia particular; ter um arquivo compartilhado significa que as mudanças feitas por qualquer membro da equipe são instantaneamente visíveis para os outros membros, mas há apenas um arquivo. Quando cópias de um arquivo são feitas, mudanças subsequentes feitas para uma cópia não afetam as outras.

• *Chamadas de sistema diversas*: A chamada chdir muda o diretório de trabalho atual. Após a chamada chdir("/usr/ast/test"); uma abertura no arquivo xyz abrirá /usr/ast/test/xyz. O conceito de um diretório de trabalho elimina a necessidade de digitar (longos) nomes de caminhos absolutos a toda hora.

A chamada de sistema kill é a maneira pela qual os usuários e os processos de usuários enviam sinais. Se um processo está preparado para capturar um sinal em particular, então, quando ele chega, uma rotina de tratamento desse sinal é executada. Se o processo não está preparado para lidar com um sinal, então sua chegada mata o processo (daí seu nome).

• *A API Win32 do Windows*: Até aqui nos concentramos fundamentalmente no UNIX. Agora chegou o momento para examinarmos com brevidade o Windows. O Windows e o UNIX diferem de uma maneira fundamental em seus respectivos modelos de programação. Um programa UNIX consiste de um código que faz uma coisa ou outra, fazendo chamadas de sistema para ter determinados serviços realizados. Em comparação, um programa Windows é normalmente direcionado por eventos. O programa principal espera por algum evento acontecer, então chama uma rotina para lidar com ele. Eventos típicos são teclas sendo pressionadas, o mouse sendo movido, um botão do mouse acionado, ou um disco flexível inserido. Tratadores são então chamados para processar o evento, atualizar a tela e o estado do programa interno.

Estrutura de sistemas operacionais

• *Sistemas monolíticos:* De longe a organização mais comum, na abordagem monolítica todo o sistema operacional é executado como um único programa em modo núcleo. O sistema operacional é escrito como uma coleção de rotinas, ligadas a um único grande

programa binário executável. Quando a técnica é usada, cada procedimento no sistema é livre para chamar qualquer outro, se este oferecer alguma computação útil de que o primeiro precisa. Ser capaz de chamar qualquer procedimento que você quer é muito eficiente, mas ter milhares de procedimentos que podem chamar um ao outro sem restrições pode também levar a um sistema difícil de lidar e compreender. Também, uma quebra em qualquer uma dessas rotinas derrubará todo o sistema operacional.

Estrutura básica para o sistema operacional:
1. Um programa principal que invoca a rotina de serviço requisitada.
2. Um conjunto de rotinas de serviço que executam as chamadas de sistema.
3. Um conjunto de rotinas utilitárias que ajudam as rotinas de serviço.

• *Sistemas de camadas:* Organizar o sistema operacional como uma hierarquia de camadas, cada uma construída sobre a camada abaixo dela.

Ex:
5 O operador
4 Programas de usuário
3 Gerenciamento de entrada/saída
2 Comunicação operador–processo
1 Memória e gerenciamento de tambor
0 Alocação do processador e multiprogramação

• *Micronúcleos:* A ideia básica por trás do projeto de micronúcleo é atingir uma alta confiabilidade através da divisão do sistema operacional em módulos pequenos e bem definidos, apenas um dos quais — o micronúcleo — é executado em modo núcleo e o resto é executado como processos de usuário comuns relativamente sem poder. Em particular, ao se executar cada driver de dispositivo e sistema de arquivos como um processo de usuário em separado, um erro em um deles pode derrubar esse componente, mas não consegue derrubar o sistema inteiro. Desse modo, um erro no driver de áudio fará que o som fique truncado ou pare, mas não derrubará o computador. Em comparação, em um sistema monolítico, com todos os drivers no núcleo, um driver de áudio com problemas pode facilmente referenciar um endereço de memória inválido e provocar uma parada dolorosa no sistema instantaneamente.

Um servidor interessante é o servidor de reencarnação, cujo trabalho é conferir se os outros servidores e drivers estão funcionando corretamente. No caso da detecção de um servidor ou driver defeituoso, ele é automaticamente substituído sem qualquer intervenção do usuário. Dessa maneira, o sistema está regenerando a si mesmo e pode atingir uma alta confiabilidade.

• *O modelo cliente-servidor:* Uma ligeira variação da ideia do micronúcleo é distinguir duas classes de processos, os servidores, que prestam algum serviço, e os clientes, que usam esses serviços. Esse modelo é conhecido como o modelo cliente-servidor.

A comunicação entre clientes e servidores é realizada muitas vezes pela troca de mensagens. Para obter um serviço, um processo cliente constrói uma mensagem dizendo o que ele quer e a envia ao serviço apropriado. O serviço então realiza o trabalho e envia de volta a resposta. Se acontecer de o cliente e o servidor serem executados na mesma máquina, determinadas otimizações são possíveis, mas conceitualmente, ainda estamos falando da troca de mensagens aqui.

• *Máquinas virtuais:* Um sistema de compartilhamento de tempo fornece (1) multiprogramação e (2) uma máquina estendida com uma interface mais conveniente do que apenas o hardware. A essência do VM/370 é separar completamente essas duas funções. O cerne do sistema, conhecido como o monitor de máquina virtual, opera direto no hardware e realiza a multiprogramação, fornecendo não uma, mas várias máquinas virtuais para a camada seguinte. No entanto, diferentemente de todos os outros sistemas operacionais, essas máquinas virtuais não são máquinas estendidas, com arquivos e outros aspectos interessantes. Em vez disso, elas são cópias exatas do hardware exposto, incluindo modos núcleo/usuário, E/S, interrupções e tudo mais que a máquina tem.

Como cada máquina virtual é idêntica ao hardware original, cada uma delas pode executar qualquer sistema operacional capaz de ser executado diretamente sobre o hardware. Máquinas virtuais diferentes podem — e frequentemente o fazem — executar diferentes sistemas operacionais.

Muitas empresas tradicionais executavam seus próprios servidores de correio, de web, de FTP e outros servidores em computadores separados, às vezes com sistemas operacionais diferentes. Elas veem a virtualização como uma maneira de executar todos eles na mesma máquina sem correr o risco de um travamento em um servidor derrubar a todos. A virtualização também é popular no mundo da hospedagem de páginas da web. Sem a virtualização, os clientes de hospedagem na web são obrigados a escolher entre a hospedagem compartilhada (que dá a eles uma conta de acesso a um servidor da web, mas nenhum controle sobre o software do servidor) e a hospedagem dedicada (que dá a eles a própria máquina, que é muito flexível, mas cara para sites de pequeno a médio porte). Quando uma empresa de hospedagem na web oferece máquinas virtuais para alugar, uma única máquina física pode executar muitas máquinas virtuais, e cada uma delas parece ser uma máquina completa. Clientes que alugam uma máquina virtual podem executar qualquer sistema operacional e software que eles quiserem, mas a uma fração do custo de um servidor dedicado (pois a mesma máquina física dá suporte a muitas máquinas virtuais ao mesmo tempo).

Outro uso da virtualização é por usuários finais que querem poder executar dois ou mais sistemas operacionais ao mesmo tempo, digamos Windows e Linux, pois alguns dos seus pacotes de aplicativos favoritos são executados em um sistema e outros no outro sistema. O termo “monitor de máquina virtual” foi renomeado como hipervisor tipo 1, que é bastante usado hoje, pois “monitor de máquina virtual” exige mais toques no teclado do que as pessoas estão preparadas para suportar agora. Observe que muitos autores usam os dois termos naturalmente.

• *Exonúcleos*: Em vez de clonar a máquina real, como é feito com as máquinas virtuais, outra estratégia é dividi-la, ou em outras palavras, dar a cada usuário um subconjunto dos recursos. Desse modo, uma máquina virtual pode obter os blocos de disco de 0 a 1.023, a próxima pode ficar com os blocos 1.024 a 2.047 e assim por diante. Na camada de baixo, executando em modo núcleo, há um programa chamado exonúcleo (ENGLER et al., 1995). Sua tarefa é alocar recursos às máquinas virtuais e então conferir tentativas de usá-las para assegurar-se de que nenhuma máquina esteja tentando usar os recursos de outra pessoa. Cada máquina virtual no nível do usuário pode executar seu próprio sistema operacional, como na VM/370 e no modo virtual 8086 do Pentium, exceto que cada uma está restrita a usar apenas os recursos que ela pediu e foram alocados. A vantagem do esquema do exonúcleo é que ele poupa uma camada de mapeamento.

O mundo de acordo com a linguagem C

Java, Python e C são todas linguagens imperativas com tipos de dados, variáveis e comandos de controle, por exemplo. Os tipos de dados primitivos em C são inteiros (incluindo curtos e longos), caracteres e números de ponto flutuante. Os tipos de dados compostos em C são similares àqueles em Java, incluindo os comandos if, switch, for e while. Funções e parâmetros são mais ou menos os mesmos em ambas as linguagens.

Pesquisa em sistemas operacionais

Virtualmente todos os pesquisadores de sistemas operacionais sabem que os sistemas operacionais atuais são enormes, inflexíveis, inconfiáveis, inseguros e carregados de erros, uns mais que os outros (os nomes não são citados aqui para proteger os culpados). Consequentemente, há muita pesquisa sobre como construir sistemas operacionais melhores.