

Aula 3

Professores:

Felipe M. G. França
Valmir C. Barbosa

Conteúdo:

Implementação

- Gerenciamento da multiprogramação
- Estruturas dos sistemas operacionais

Gerenciamento da Multiprogramação

→ Pelo sistema operacional:

- O sistema operacional gerencia a alocação do processador entre os processos.
- A multiprogramação é visível ao usuário.

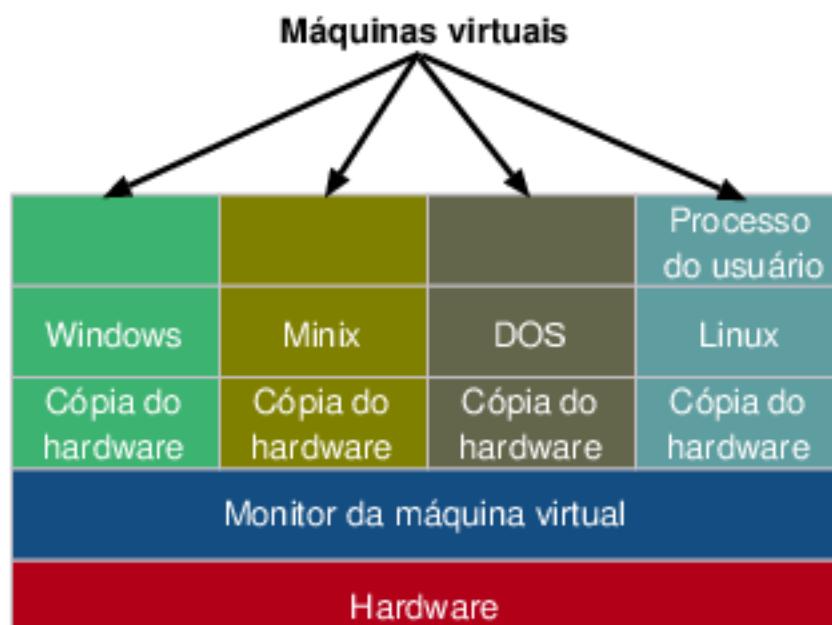
→ Por uma extensão do sistema operacional:

- O sistema operacional trata essencialmente de fornecer uma máquina estendida ao usuário.
- A multiprogramação não é visível ao usuário.
- Permite a separação dos conceitos de multiprogramação e de máquina estendida.

→ Vamos estudar brevemente dois sistemas para o gerenciamento da multiprogramação: **Máquinas virtuais** e **Exonúcleo**.

Máquinas virtuais

- ▶ Define-se um conjunto de **máquinas virtuais**, que são cópias idênticas ao hardware real, e não máquinas estendidas.
- ▶ O principal conceito é o do **monitor de máquina virtual**, um programa executando sobre o hardware da máquina:

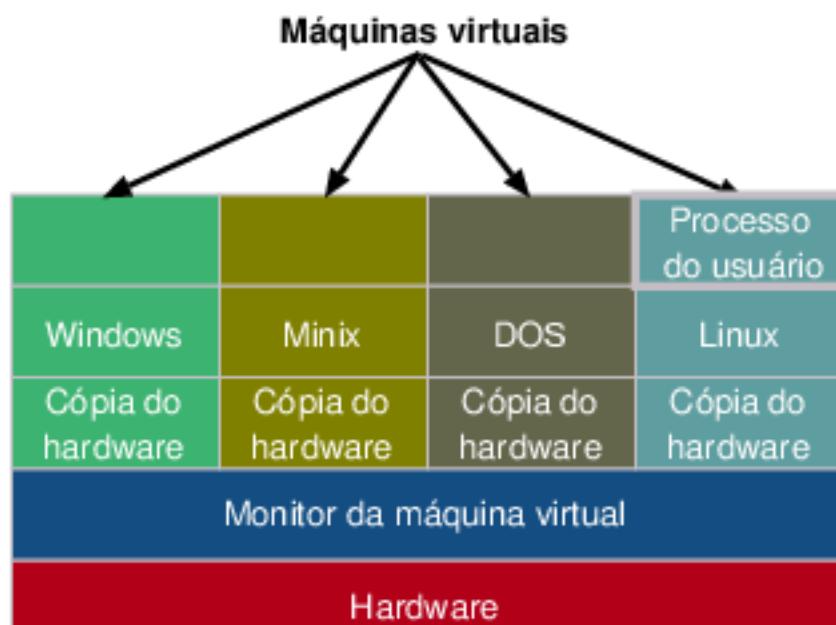


O monitor de máquina virtual cria máquinas virtuais que são cópias exatas do hardware do computador, e estas máquinas virtuais não são estendidas (os dispositivos não são abstratos). Todo sistema operacional que executa em uma das máquinas virtuais acha que está executando sobre o hardware real da máquina.



Máquinas virtuais

- ▶ Define-se um conjunto de **máquinas virtuais**, que são cópias idênticas ao hardware real, e não máquinas estendidas.
- ▶ O principal conceito é o do **monitor de máquina virtual**, um programa executando sobre o hardware da máquina:

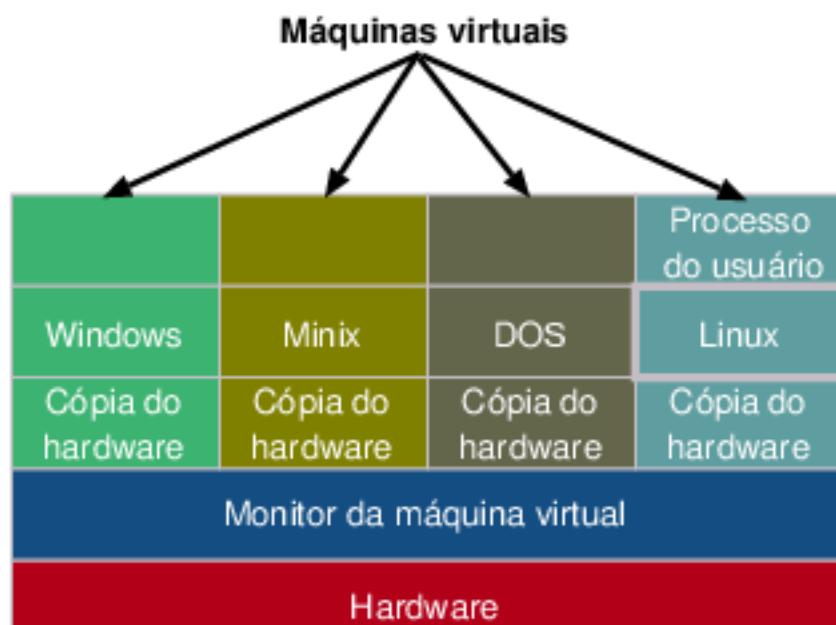


Um processo do usuário executando sobre o Linux em uma das máquinas virtuais precisa de algum serviço do sistema, como o de abrir um arquivo, e chamou um procedimento da biblioteca, que por sua vez fez a chamada ao sistema operacional para abrir o arquivo.



Máquinas virtuais

- ▶ Define-se um conjunto de **máquinas virtuais**, que são cópias idênticas ao hardware real, e não máquinas estendidas.
- ▶ O principal conceito é o do **monitor de máquina virtual**, um programa executando sobre o hardware da máquina:

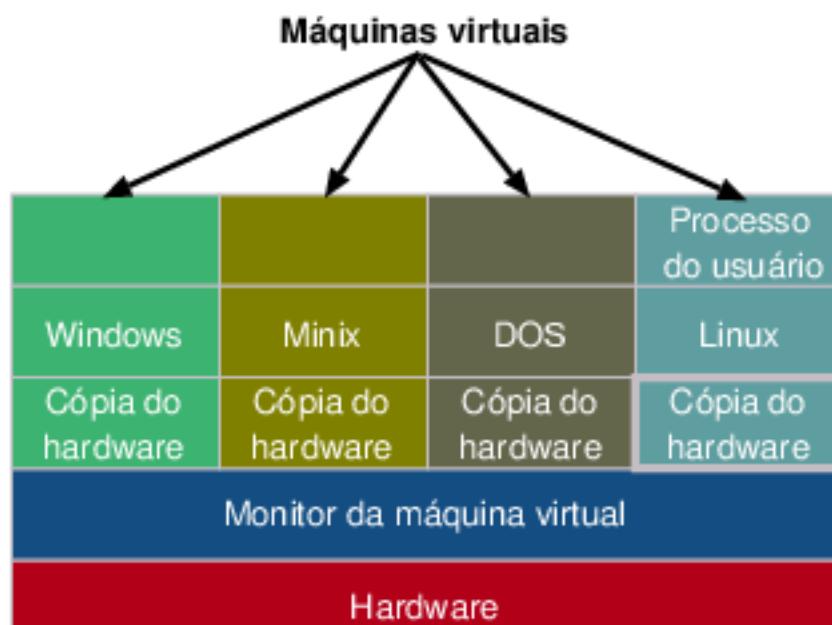


A chamada ao sistema para abrir o arquivo eventualmente acessará o disco abstrato criado pelo sistema operacional. Com disso, a parte do sistema operacional que trata do disco abstrato precisará acessar o disco da máquina virtual, achando que está na verdade acessando o disco real do computador.



Máquinas virtuais

- ▶ Define-se um conjunto de **máquinas virtuais**, que são cópias idênticas ao hardware real, e não máquinas estendidas.
- ▶ O principal conceito é o do **monitor de máquina virtual**, um programa executando sobre o hardware da máquina:

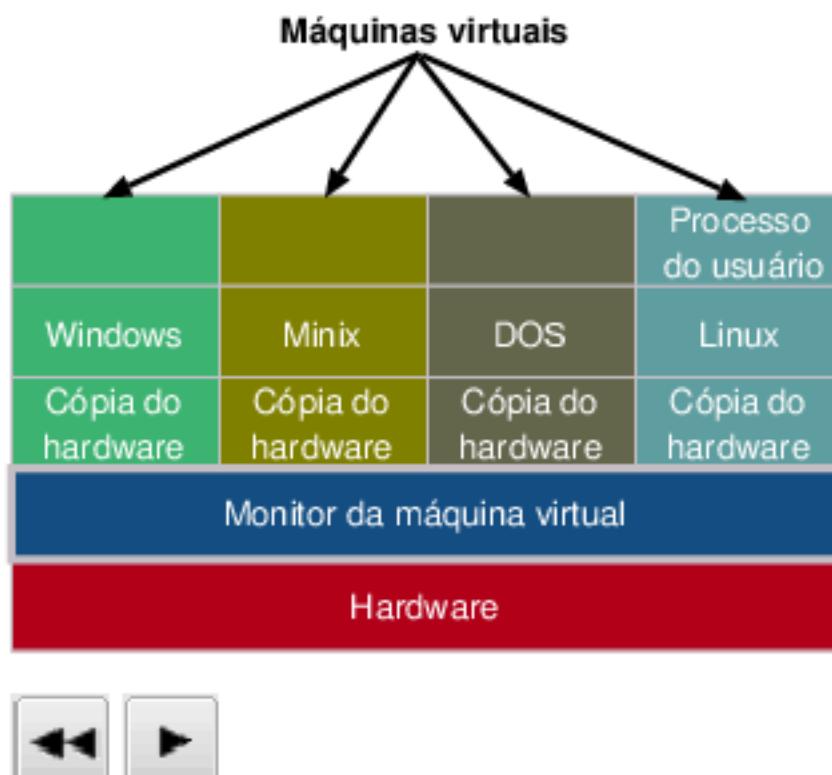


O acesso ao disco simulado pela máquina virtual causará então uma chamada ao monitor da máquina virtual, para que este possa acessar o disco real do hardware, pois o acesso ao disco é transparente ao sistema operacional Linux executando sobre a máquina virtual.



Máquinas virtuais

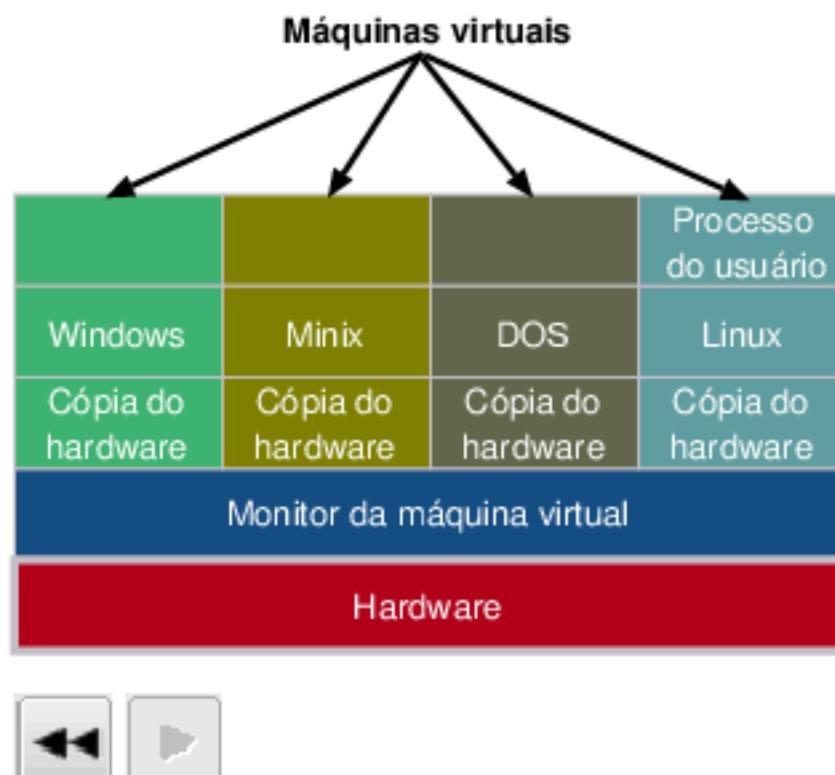
- ▶ Define-se um conjunto de **máquinas virtuais**, que são cópias idênticas ao hardware real, e não máquinas estendidas.
- ▶ O principal conceito é o do **monitor de máquina virtual**, um programa executando sobre o hardware da máquina:



O monitor da máquina virtual então acessa o disco real do computador, para simular um disco para a máquina virtual. Para isso, o monitor acessa diretamente o disco real, como faria o Linux se este executasse sobre o hardware real da máquina.

Máquinas virtuais

- ▶ Define-se um conjunto de **máquinas virtuais**, que são cópias idênticas ao hardware real, e não máquinas estendidas.
- ▶ O principal conceito é o do **monitor de máquina virtual**, um programa executando sobre o hardware da máquina:



Depois de o monitor da máquina virtual enviar ao disco rígido do computador os comandos para ler os dados necessários à máquina virtual, o disco é finalmente acessado, e lê estes dados. Depois disso, o monitor passa estes dados ao Linux, que acha que estes foram lidos do disco real do computador.

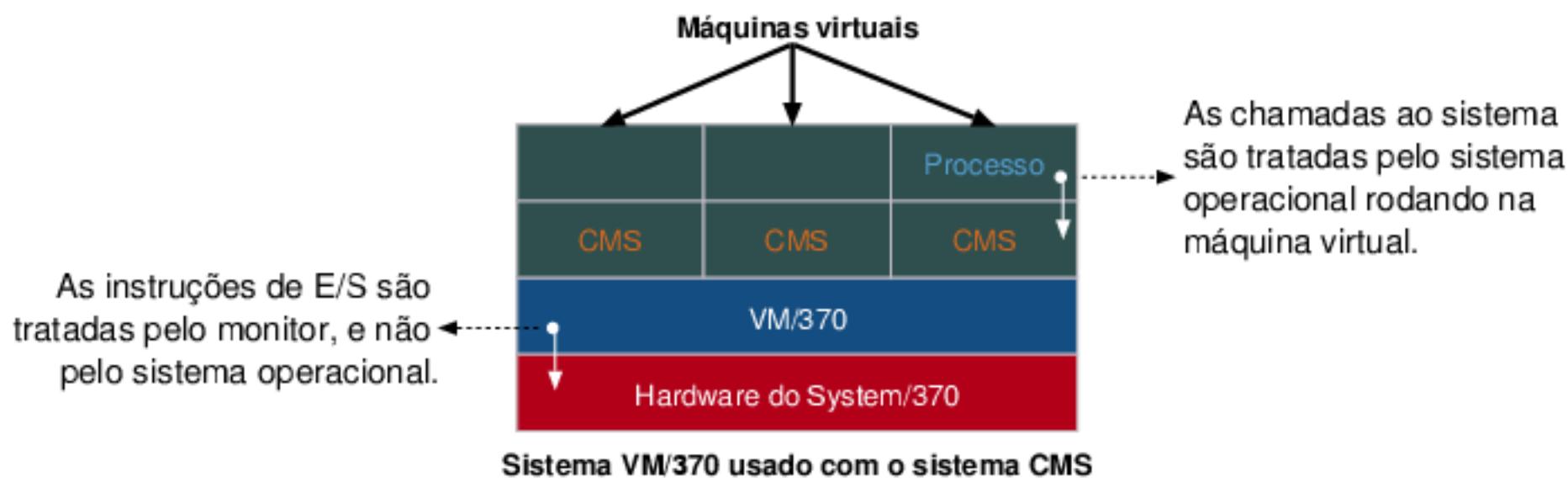
Máquinas virtuais

- ➡ Então, o monitor de máquina virtual:
 - ➡ Define um conjunto de máquinas virtuais que são cópias exatas do hardware, e não máquinas estendidas.
 - ➡ Cada máquina pode executar um sistema operacional diferente.
 - ➡ Somente o acesso ao hardware é tratado pelo monitor.
 - ➡ A multiprogramação é tratada pelo monitor de maneira natural, ao gerenciar as máquinas virtuais.
- ➡ Um monitor de máquina virtual também pode ser implementado em hardware, como no processador Pentium.
- ➡ Existem outros softwares que criam máquinas virtuais, como o Bochs, o Qemu, o Virtual PC, e o VMWare.
- ➡ O conceito de máquina virtual também é usado em outras áreas, como a máquina JVM que interpreta os programas em JAVA.

Máquinas virtuais

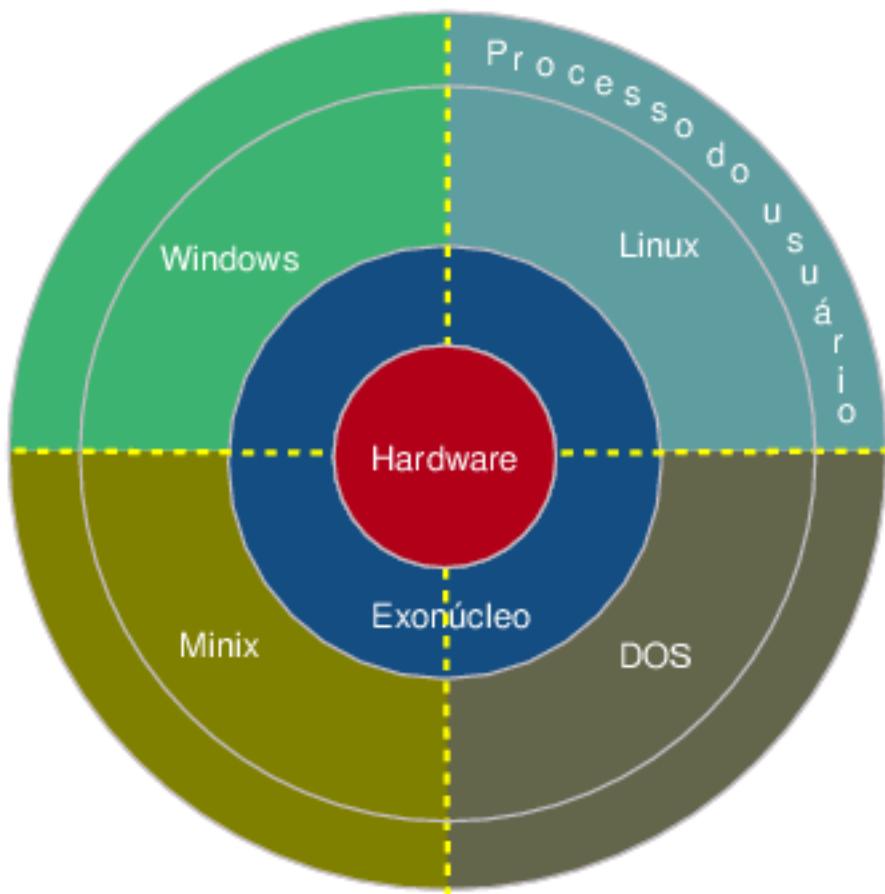
→ O VM/370 é um exemplo de um monitor de máquina virtual:

- Executava sobre o computador de grande porte System/370.
- Desenvolvido para sistemas de compartilhamento de tempo.
- Cada máquina virtual executava, em geral, o sistema CMS para o compartilhamento de tempo.



Exonúcleo

- Neste caso todos os recursos do hardware são divididos entre as máquinas virtuais.
- O programa executando sobre o hardware é chamado de **exonúcleo**:

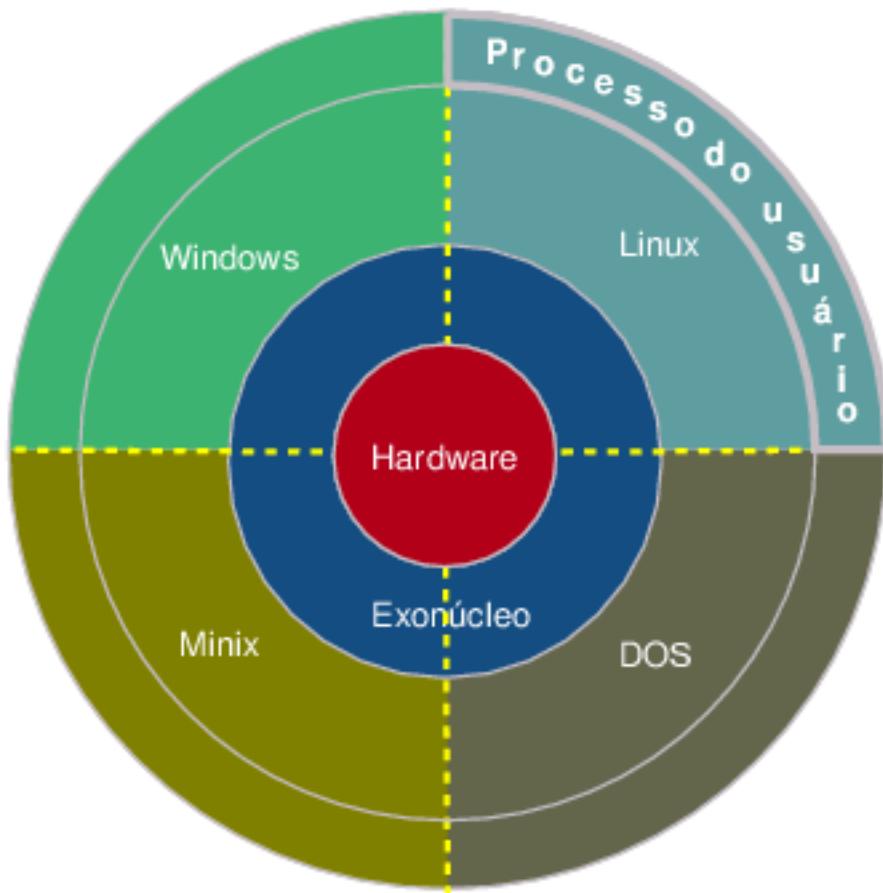


O exonúcleo, ao invés de criar máquinas virtuais que são cópias exatas do hardware, divide os recursos do hardware entre todas as máquinas virtuais, e evita que uma máquina virtual acesse indevidamente as partes dos recursos associados às outras máquinas virtuais. Nesta implementação, o sistema operacional executando sobre a máquina virtual sabe do compartilhamento do hardware.



Exonúcleo

- ➡ Neste caso todos os recursos do hardware são divididos entre as máquinas virtuais.
- ➡ O programa executando sobre o hardware é chamado de **exonúcleo**:

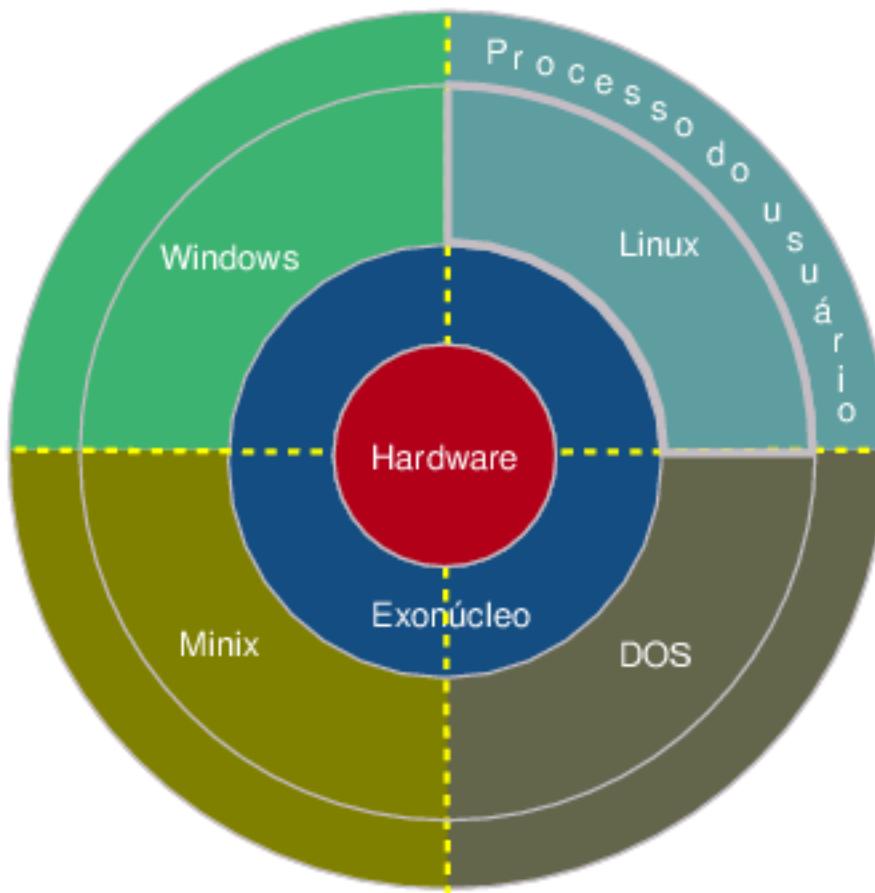


Assim como antes, um processo do usuário estava executando, e precisou abrir um arquivo. Com isso, o processo chamou a função da biblioteca para abrir o arquivo, que por sua vez fez uma chamada ao sistema operacional para abrir o arquivo.



Exonúcleo

- ➡ Neste caso todos os recursos do hardware são divididos entre as máquinas virtuais.
- ➡ O programa executando sobre o hardware é chamado de **exonúcleo**:

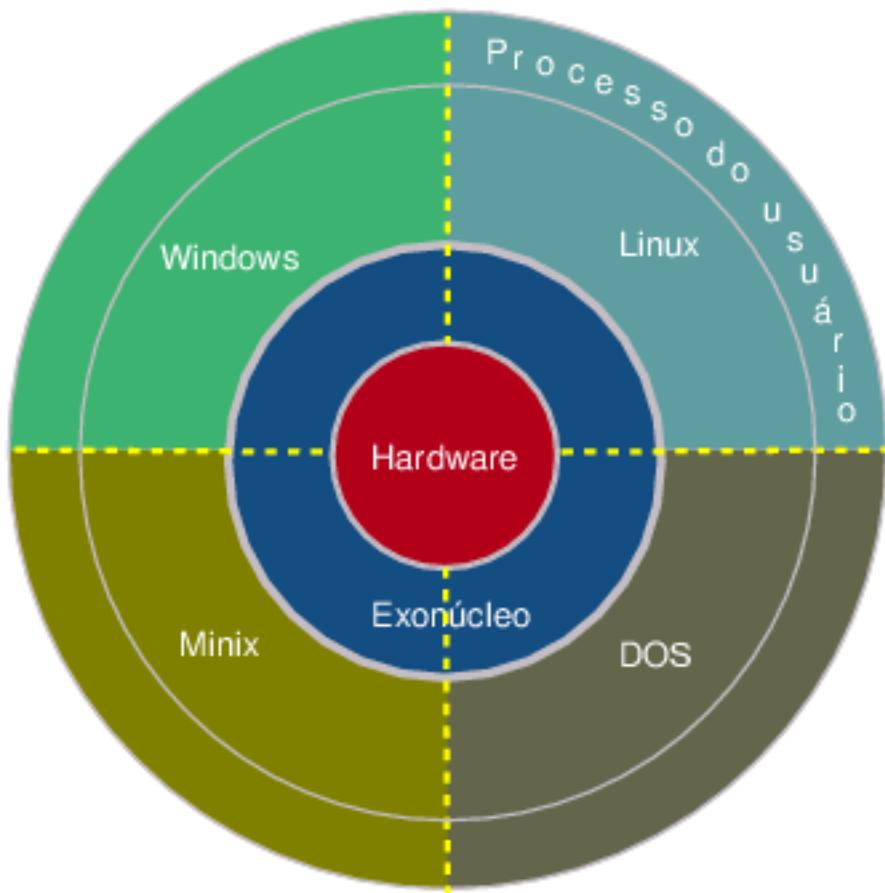


A chamada ao sistema causa o desvio para o código do núcleo que trata da abertura de um arquivo. Este código deverá então acessar o disco abstrato para obter as informações sobre o arquivo a ser aberto. A parte do sistema que trata deste disco abstrato acessará a parte do disco real do computador alocado à máquina virtual em que o Linux está executando.



Exonúcleo

- Neste caso todos os recursos do hardware são divididos entre as máquinas virtuais.
- O programa executando sobre o hardware é chamado de **exonúcleo**:

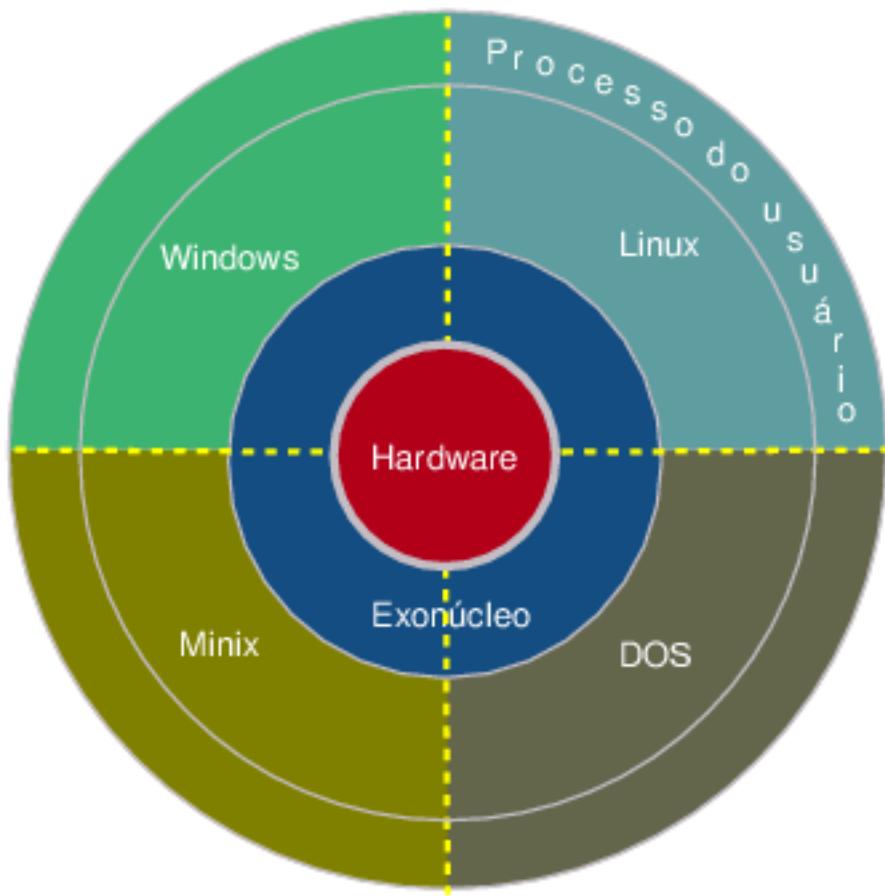


Como os sistemas operacionais devem saber do compartilhamento do hardware, o exonúcleo somente verifica se a máquina virtual está acessando a parte do disco real alocado a ela. Se a máquina virtual somente acessou esta parte, então o exonúcleo irá acessar o hardware do computador, para ler a região do disco associada a esta máquina virtual.



Exonúcleo

- ➡ Neste caso todos os recursos do hardware são divididos entre as máquinas virtuais.
- ➡ O programa executando sobre o hardware é chamado de **exonúcleo**:



Depois de o exonúcleo enviar os comandos ao disco rígido para ler as informações da parte do disco associada à máquina virtual que está executando o Linux, o dispositivo procura por estas informações, e começa a ler-las. Depois de as informações serem lidas, estas serão repassadas, pelo exonúcleo, ao Linux.



Exonúcleo

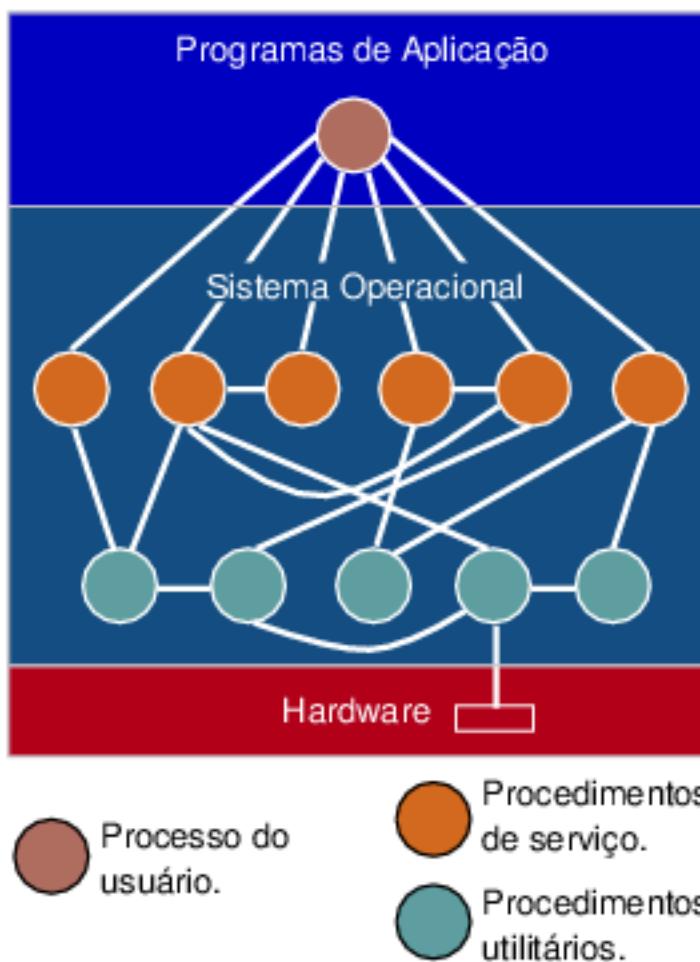
- ➡ O monitor, que é chamado de **exonúcleo**:
 - Gerencia a alocação dos recursos às máquinas virtuais.
 - Garante que cada máquina virtual use somente os recursos alocados a ela.
- ➡ Podemos também executar vários sistemas operacionais, mas estes sabem que o hardware está sendo compartilhado.
- ➡ Vantagens sobre as máquinas virtuais:
 - Gerenciamento mais simplificado das máquinas virtuais, pois os sistemas sabem que receberam uma parte dos recursos.
 - O monitor não precisa se preocupar com o gerenciamento da abstração de uma máquina virtual.
 - O gerenciamento da multiprogramação é facilitado.

Estruturas dos sistemas operacionais

- ▶ Sistemas monolíticos.
- ▶ Sistemas em camadas.
- ▶ Modelo cliente-servidor.

Sistemas monolíticos

- Não existe nenhuma organização dentro do núcleo do sistema operacional:

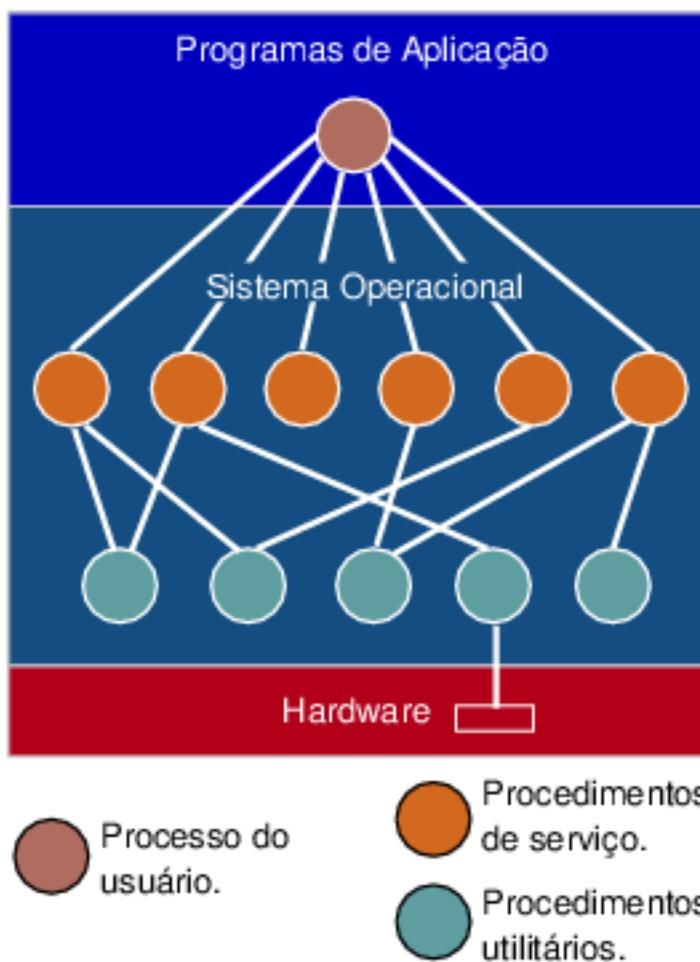


Um sistema operacional monolítico caracteriza-se por não possuir nenhuma estruturação interna no seu núcleo. O núcleo do sistema é composto por um conjunto de procedimentos, compilados em um único arquivo, que podem chamar uns aos outros. Alguns procedimentos implementam as chamadas ao sistema operacional, e outros são procedimentos utilitários, usados para executar tarefas comuns a estas chamadas.



Sistemas monolíticos

- Não existe nenhuma organização dentro do núcleo do sistema operacional:

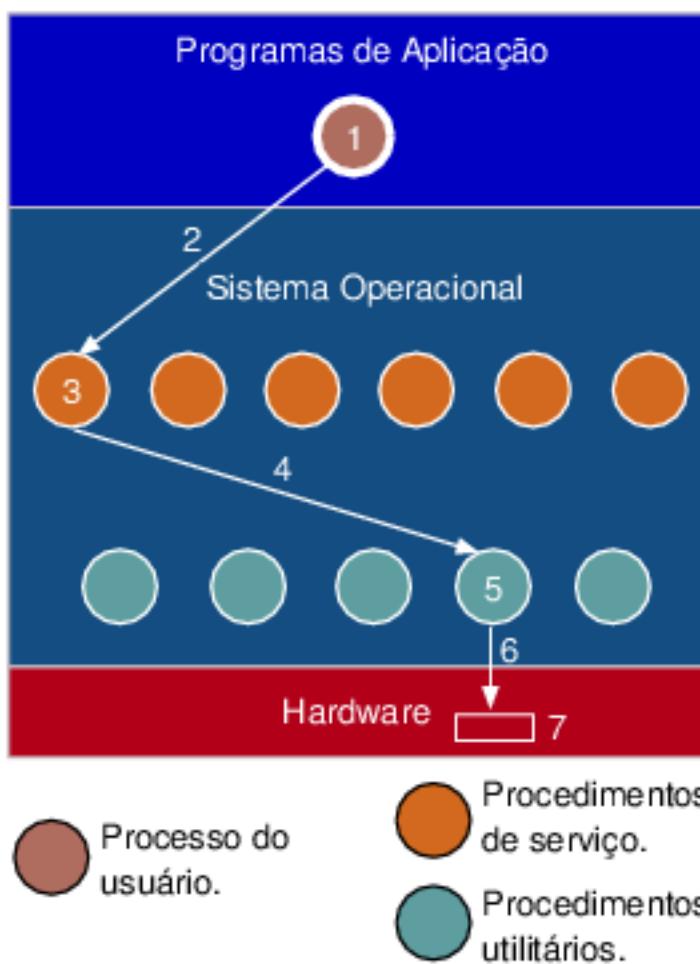


As chamadas ao sistema são feitas de modo similar ao que já vimos no curso. Este modo de chamada permite definir uma estruturação mínima ao sistema: um **processo** do usuário que usa um conjunto de **procedimentos de serviço**, que implementam as chamadas ao sistema, que por sua vez usam um conjunto de **procedimentos utilitários** para executar as tarefas comuns as chamadas ao sistema.



Sistemas monolíticos

→ Não existe nenhuma organização dentro do núcleo do sistema operacional:

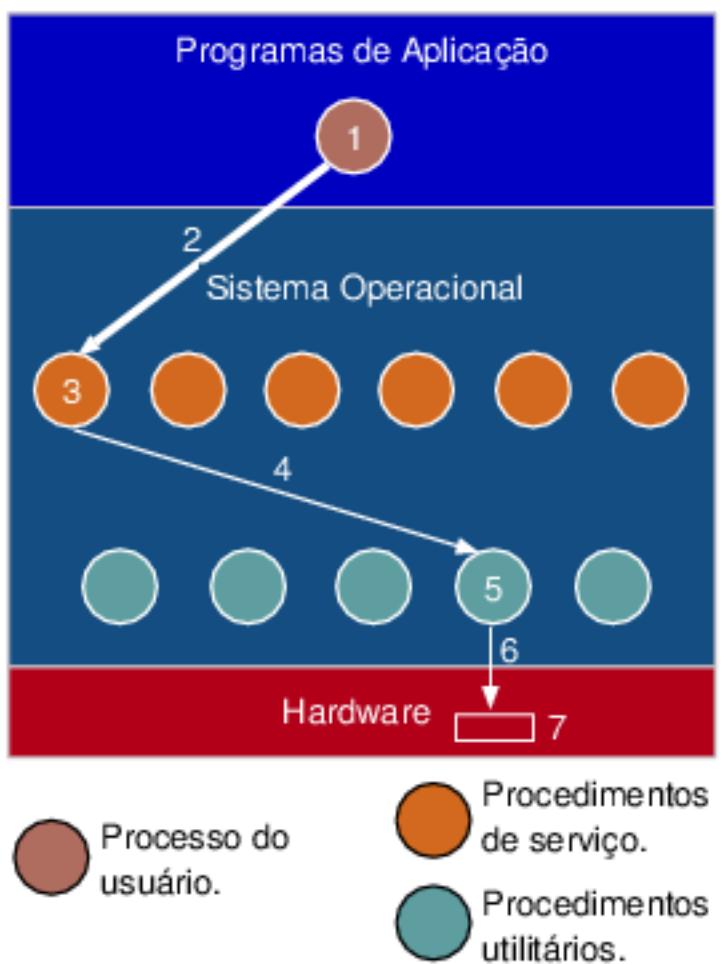


1: O processo do usuário precisou acessar um arquivo no disco. Então, este processo chamou um procedimento da biblioteca para abrir o arquivo. Como vimos, este será o procedimento que irá executar a chamada ao sistema.



Sistemas monolíticos

→ Não existe nenhuma organização dentro do núcleo do sistema operacional:

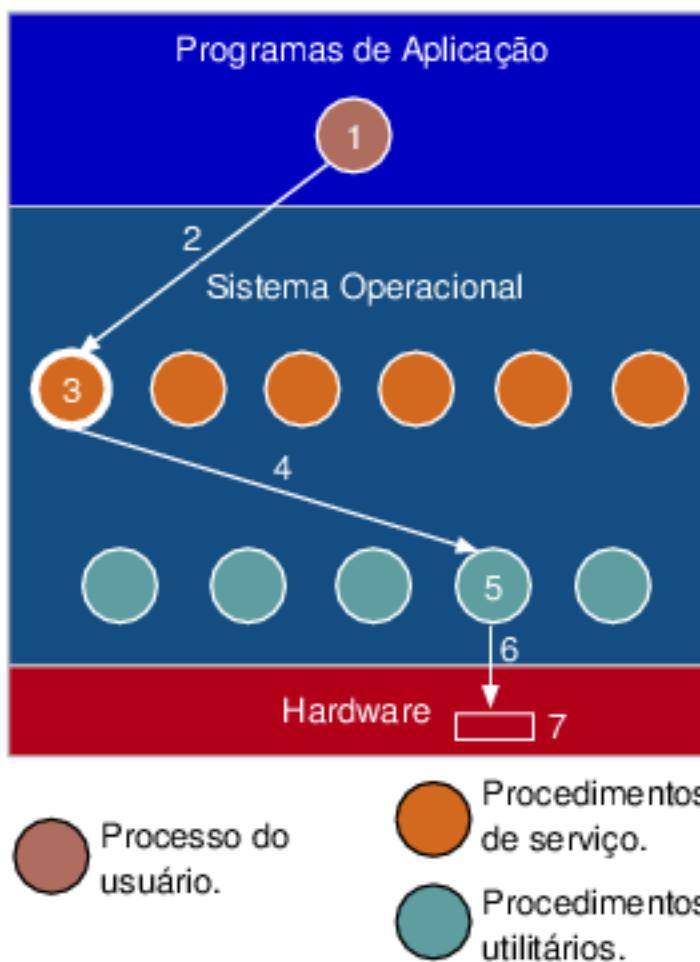


2: A biblioteca executou a instrução TRAP, e o código do núcleo do sistema chamou o tratador da chamada que abre um arquivo.



Sistemas monolíticos

→ Não existe nenhuma organização dentro do núcleo do sistema operacional:

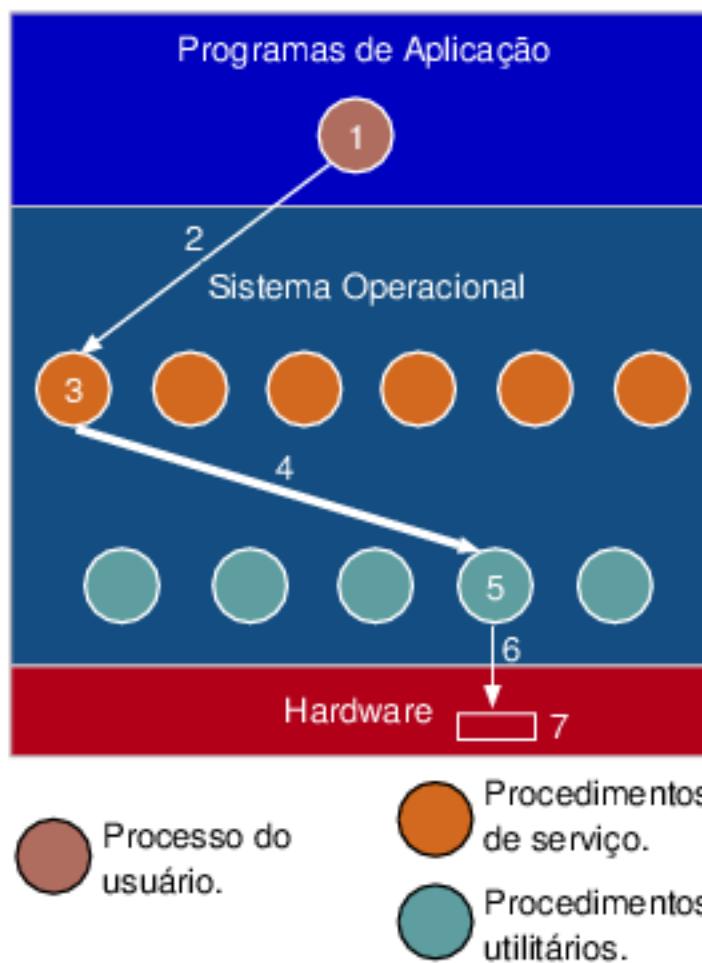


3: O tratador da chamada para ler um arquivo começa a executar o código que abre um arquivo. Depois de algum tempo, o tratador precisou acessar o disco para obter as informações sobre o arquivo. Para poder acessar o disco, o tratador precisa chamar um dos procedimentos utilizários do núcleo.



Sistemas monolíticos

→ Não existe nenhuma organização dentro do núcleo do sistema operacional:

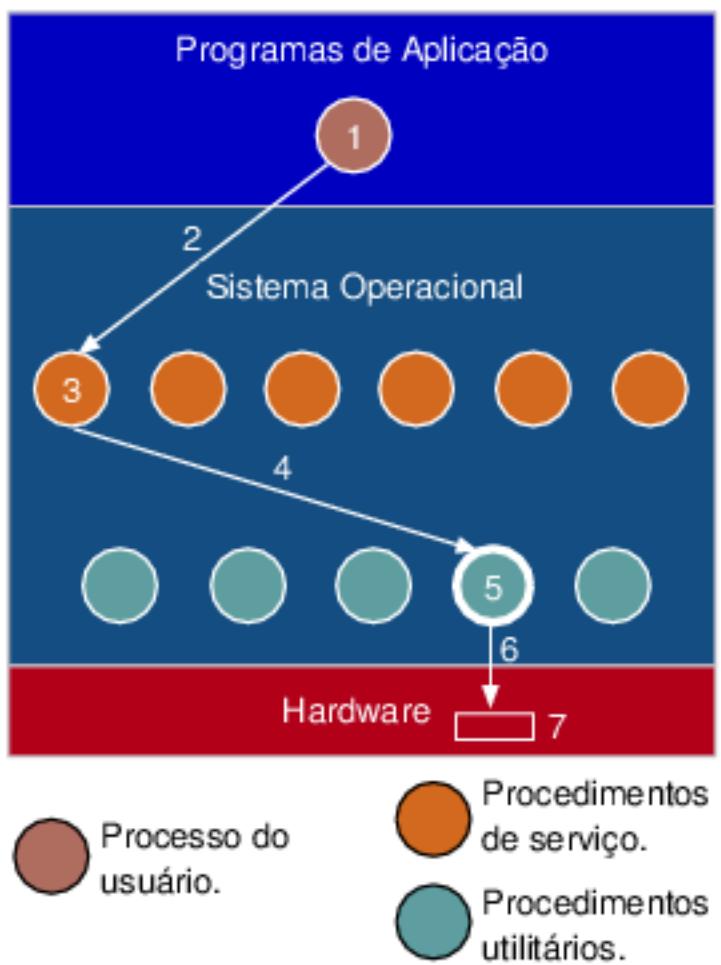


4: O procedimento utilitário para acesso ao disco é chamado pelo tratador da chamada.



Sistemas monolíticos

→ Não existe nenhuma organização dentro do núcleo do sistema operacional:

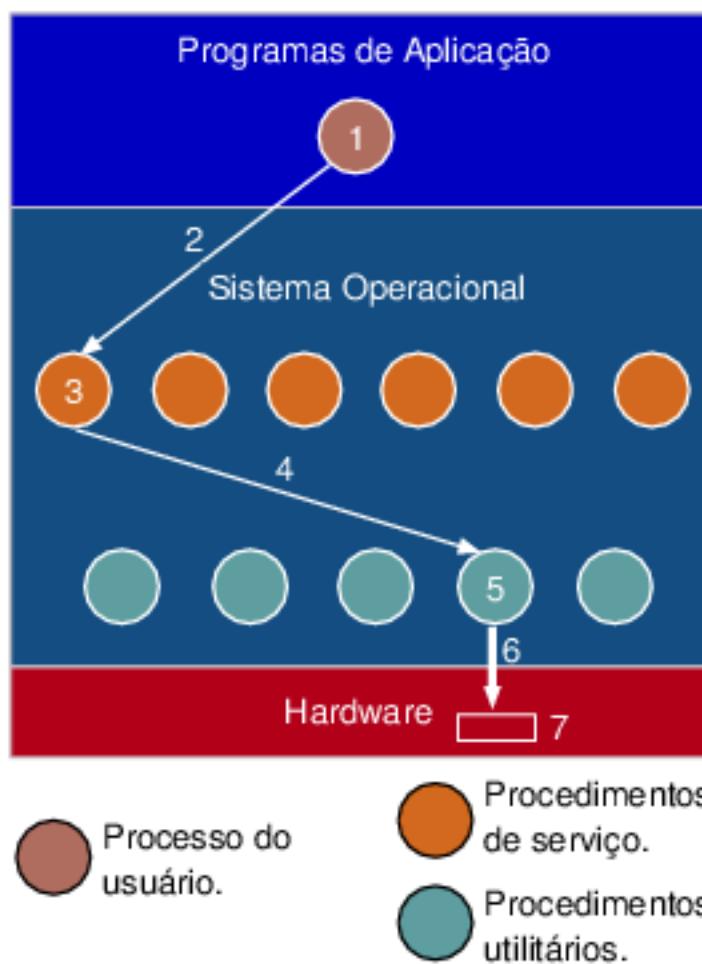


5: O procedimento utilitário para acesso ao disco começa a executar o seu código, e se prepara para acessar a região do disco com as informações do arquivo solicitadas pelo tratador da chamada.



Sistemas monolíticos

→ Não existe nenhuma organização dentro do núcleo do sistema operacional:

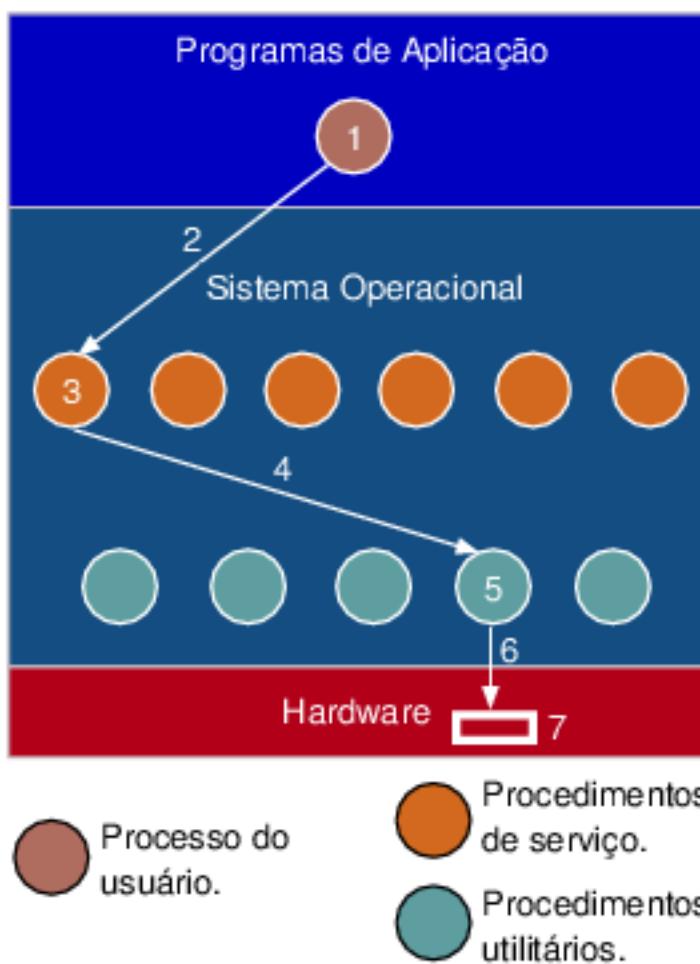


6: O procedimento utilitário para acessar o disco começa uma operação de E/S, com o objetivo de acessar as informações do arquivo que estão no disco.



Sistemas monolíticos

→ Não existe nenhuma organização dentro do núcleo do sistema operacional:

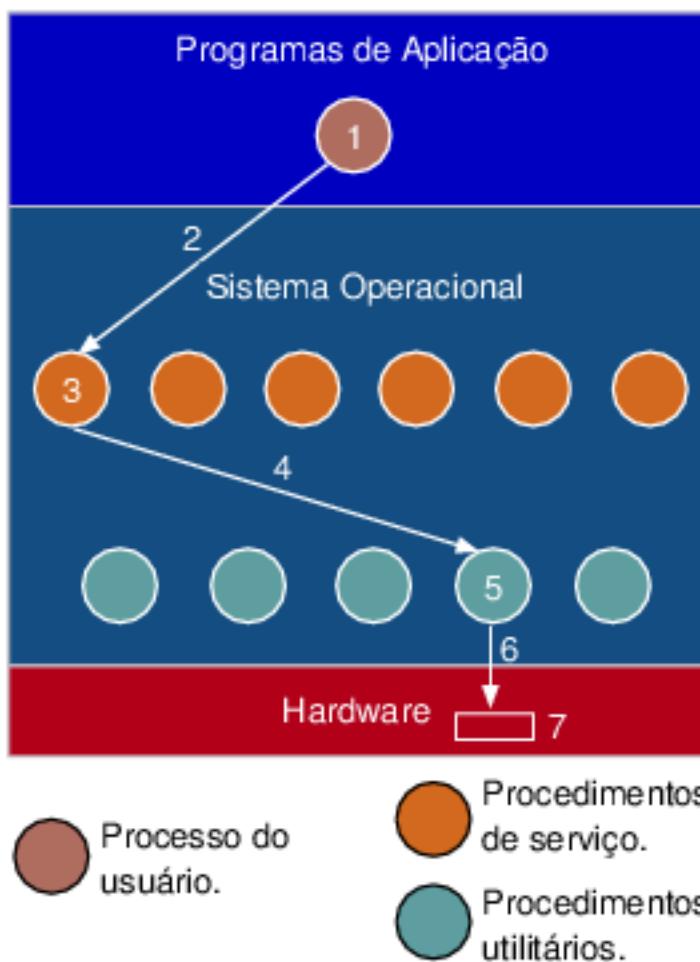


7: O disco rígido do computador é acionado, e os mecanismos do dispositivo tratam de achar e de retornar as informações da parte do disco em que estas estão gravadas.



Sistemas monolíticos

→ Não existe nenhuma organização dentro do núcleo do sistema operacional:



Depois de o disco rígido ler as informações do arquivo, o controle será retornado ao procedimento utilitário que acessa o disco. Este procedimento finalizará, e o controle será retornado para a chamada ao sistema. Depois, o controle voltará ao procedimento da biblioteca, que eventualmente retornará o controle ao processo do usuário, que continuará com a sua execução.



Sistemas monolíticos

→ O projeto do sistema não é estruturado:

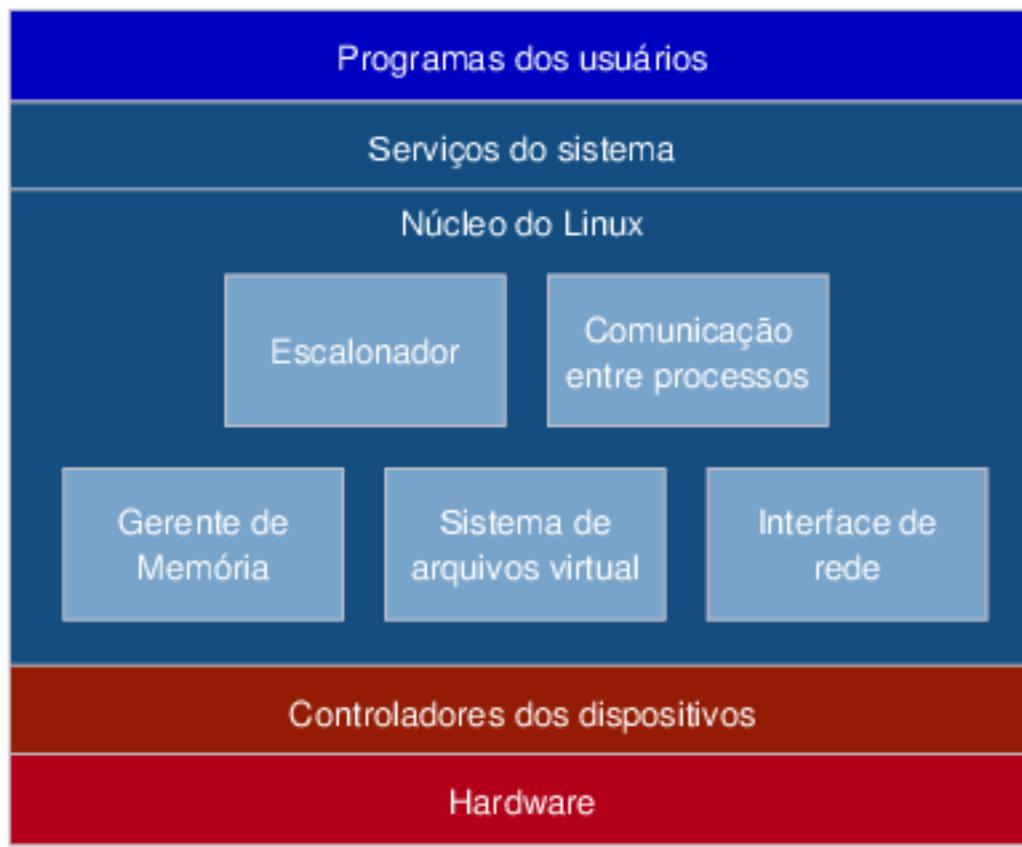
- O sistema é um conjunto de procedimentos compilados num único arquivo objeto.
- Os procedimentos possuem uma interface bem definida.
- Os procedimentos podem chamar uns aos outros.
- Os procedimentos dependem da implementação dos outros para funcionar.

→ O modelo de chamadas ao sistema permite uma estruturação:

- Um processo que usa os procedimentos de serviço.
- Um conjunto de procedimentos de serviço que implementam as chamadas ao sistema.
- Um conjunto de procedimentos utilitários usados pelos procedimentos de serviço.

Sistemas monolíticos

- O núcleo do Linux é um exemplo de um sistema monolítico:
- O acesso ao hardware é feito a partir dos controladores de dispositivos.
 - É dividido em cinco subsistemas.



O núcleo do Linux é monolítico, e possui cinco subsistemas principais:

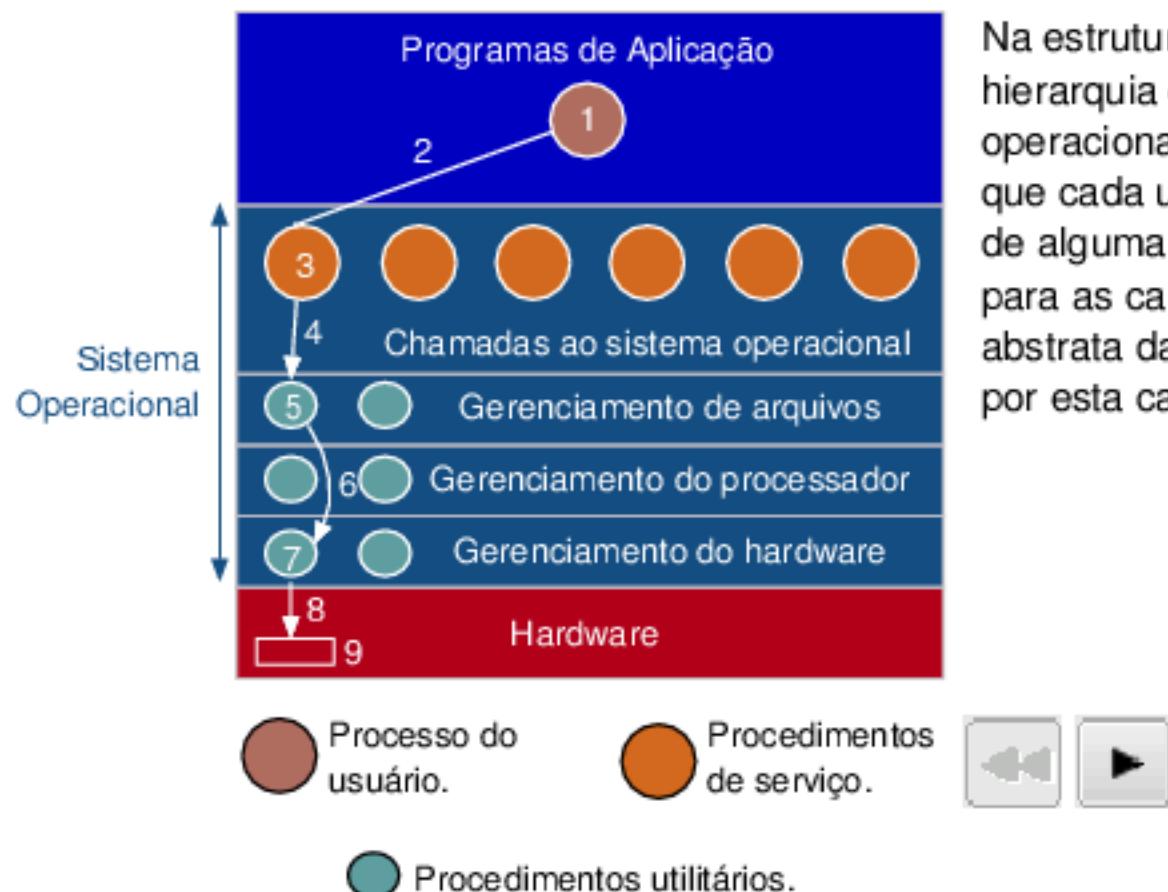
- **Escalonador;**
- **Comunicação entre processos;**
- **Gerente de Memória;**
- **Sistema de arquivos virtual;**
- **Interface de Rede;**

Para ver resumidamente a tarefa de cada subsistema, pressione o mouse sobre este subsistema.



Sistema em camadas

→ O sistema operacional é estruturado como uma hierarquia em camadas, onde cada camada implementa uma parte do sistema:

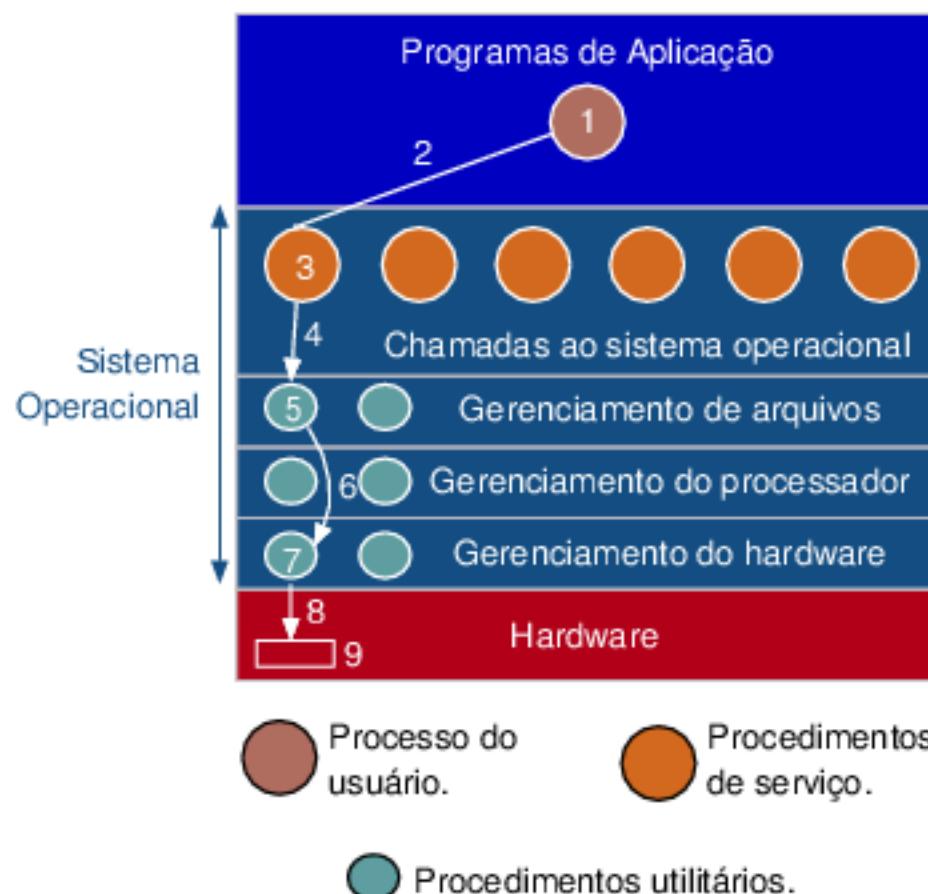


Na estruturação do sistema como uma hierarquia em camadas, o núcleo do sistema operacional é dividido em camadas, sendo que cada uma delas trata do gerenciamento de alguma parte do hardware, fornecendo para as camadas superiores uma versão abstrata da parte do hardware gerenciada por esta camada.



Sistema em camadas

→ O sistema operacional é estruturado como uma hierarquia em camadas, onde cada camada implementa uma parte do sistema:



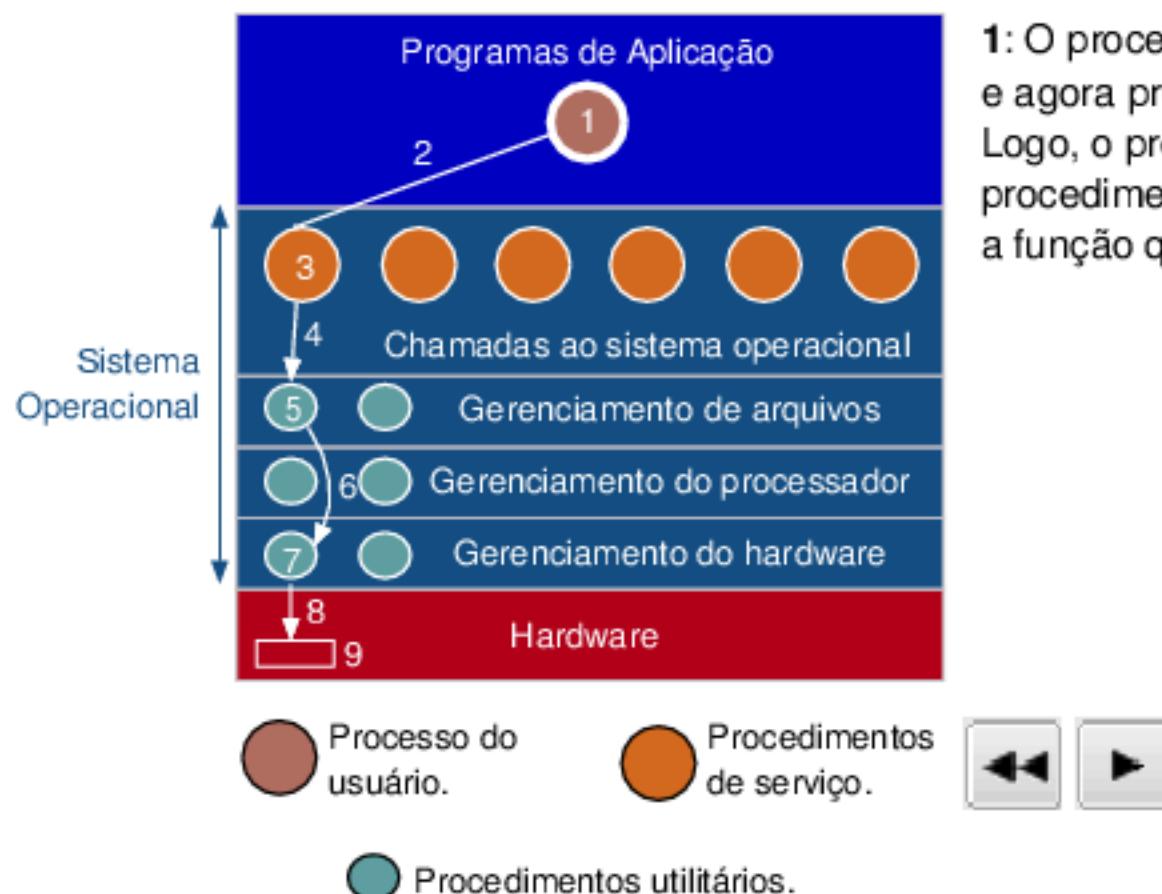
No nosso sistema exemplo, o sistema pode ser estruturado em quatro camadas:

- **Chamadas ao sistema operacional**, que implementa as chamadas ao sistema.
- **Gerenciamento de arquivos**, que cuida da implementação do sistema de arquivos.
- **Gerenciamento do processador**, que disponibiliza um ambiente abstrato em que cada processo pode rodar seqüencialmente.
- **Gerenciamento do hardware**, que fornece dispositivos abstratos mais fáceis de serem usados que os dispositivos físicos reais.



Sistema em camadas

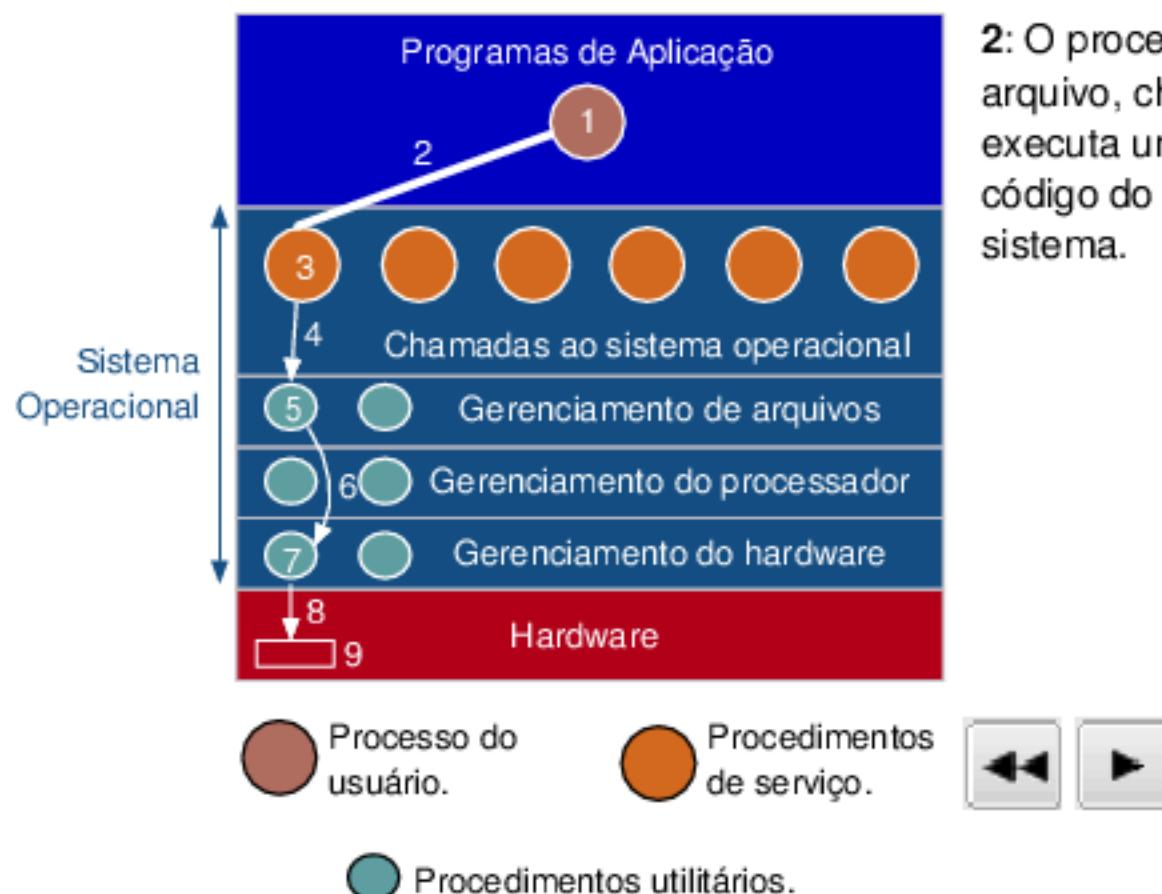
→ O sistema operacional é estruturado como uma hierarquia em camadas, onde cada camada implementa uma parte do sistema:



1: O processo do usuário estava executando, e agora precisa acessar um arquivo no disco. Logo, o processo deve chamar o procedimento da biblioteca que implementa a função que abre um arquivo no disco.

Sistema em camadas

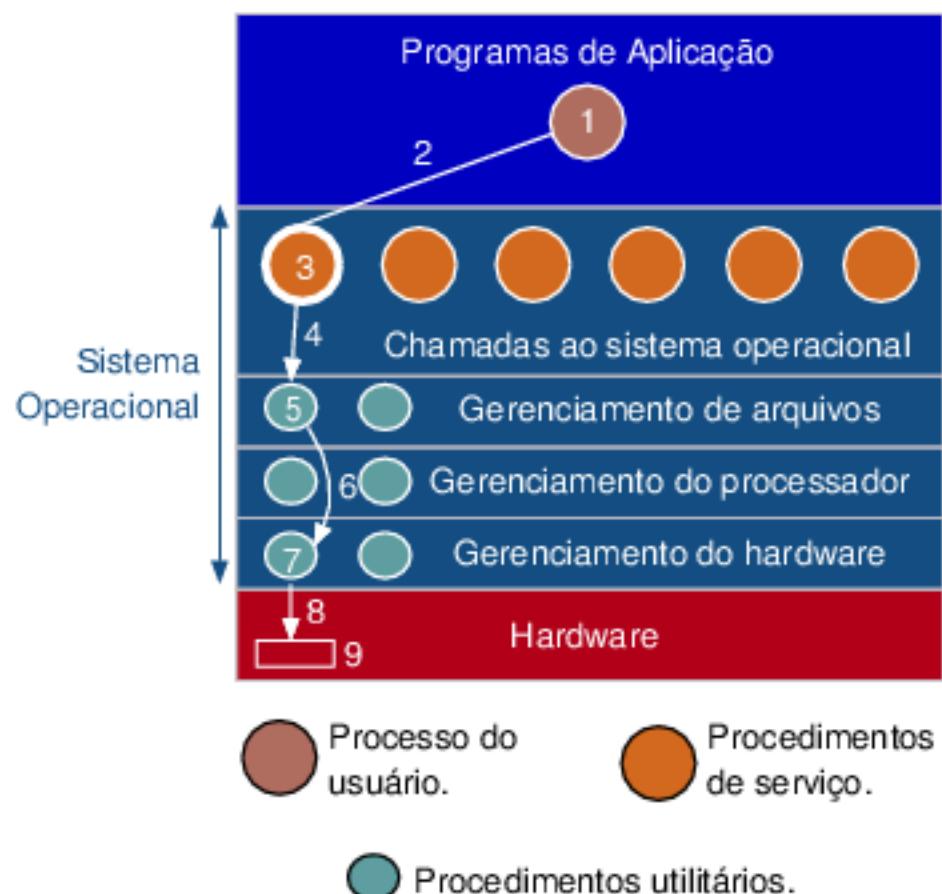
→ O sistema operacional é estruturado como uma hierarquia em camadas, onde cada camada implementa uma parte do sistema:



2: O procedimento da biblioteca para abrir o arquivo, chamado pelo processo do usuário, executa uma instrução TRAP para chamar o código do núcleo que trata das chamadas ao sistema.

Sistema em camadas

→ O sistema operacional é estruturado como uma hierarquia em camadas, onde cada camada implementa uma parte do sistema:

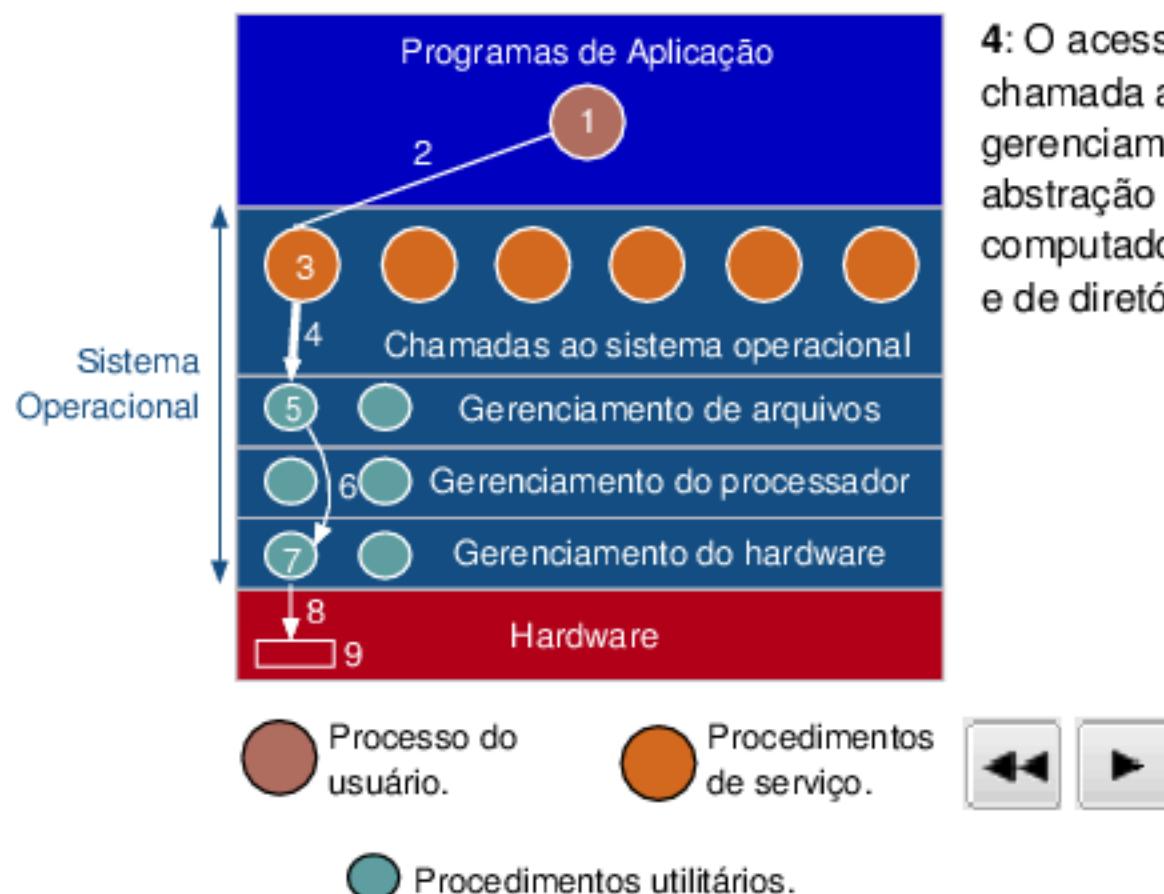


3: Este código chama o procedimento que trata da chamada ao sistema para abrir um arquivo. O procedimento tratador deverá agora obter as informações sobre o arquivo a ser aberto. Para isso, o tratador acessa o sistema de arquivos, com as informações de todos os arquivos do sistema.



Sistema em camadas

→ O sistema operacional é estruturado como uma hierarquia em camadas, onde cada camada implementa uma parte do sistema:

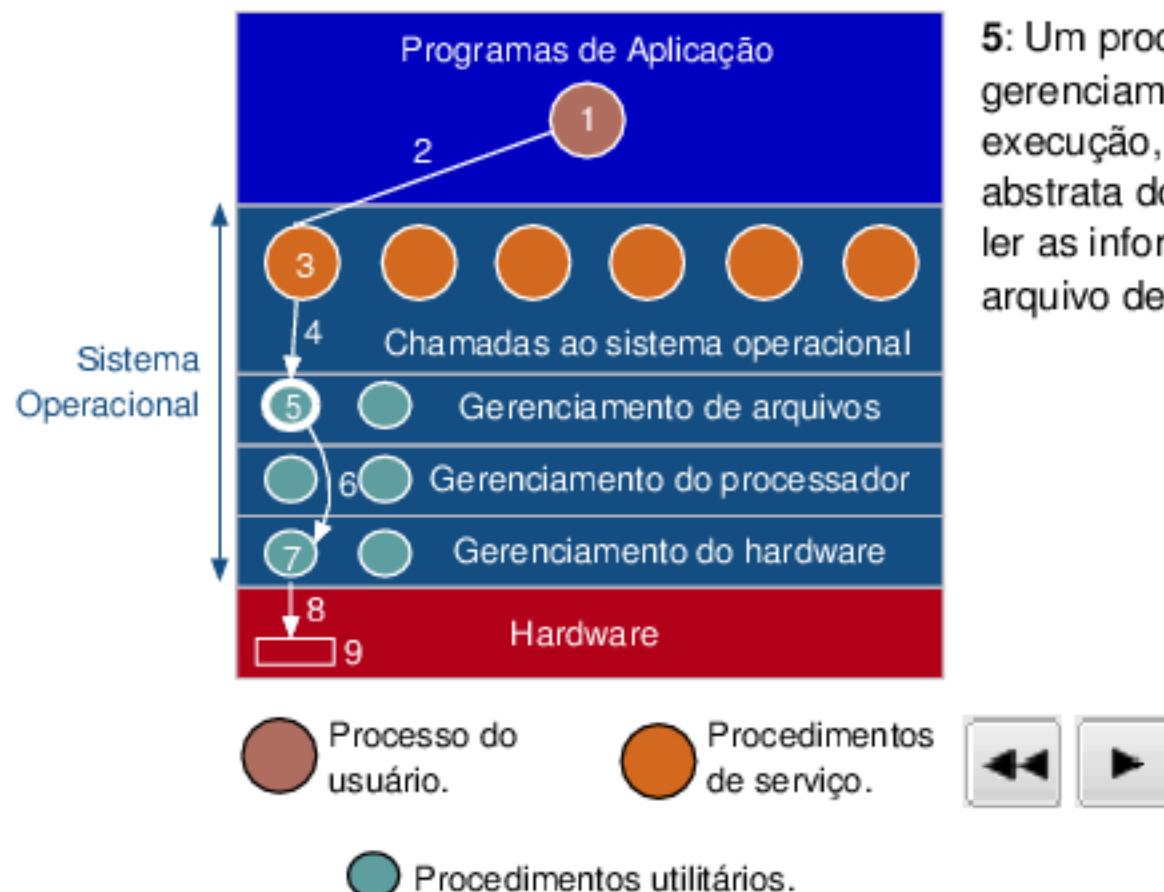


4: O acesso ao sistema de arquivos causa a chamada a um procedimento da camada de gerenciamento de arquivos, que trata da abstração que permite vermos o disco do computador como um conjunto de arquivos e de diretórios.



Sistema em camadas

→ O sistema operacional é estruturado como uma hierarquia em camadas, onde cada camada implementa uma parte do sistema:

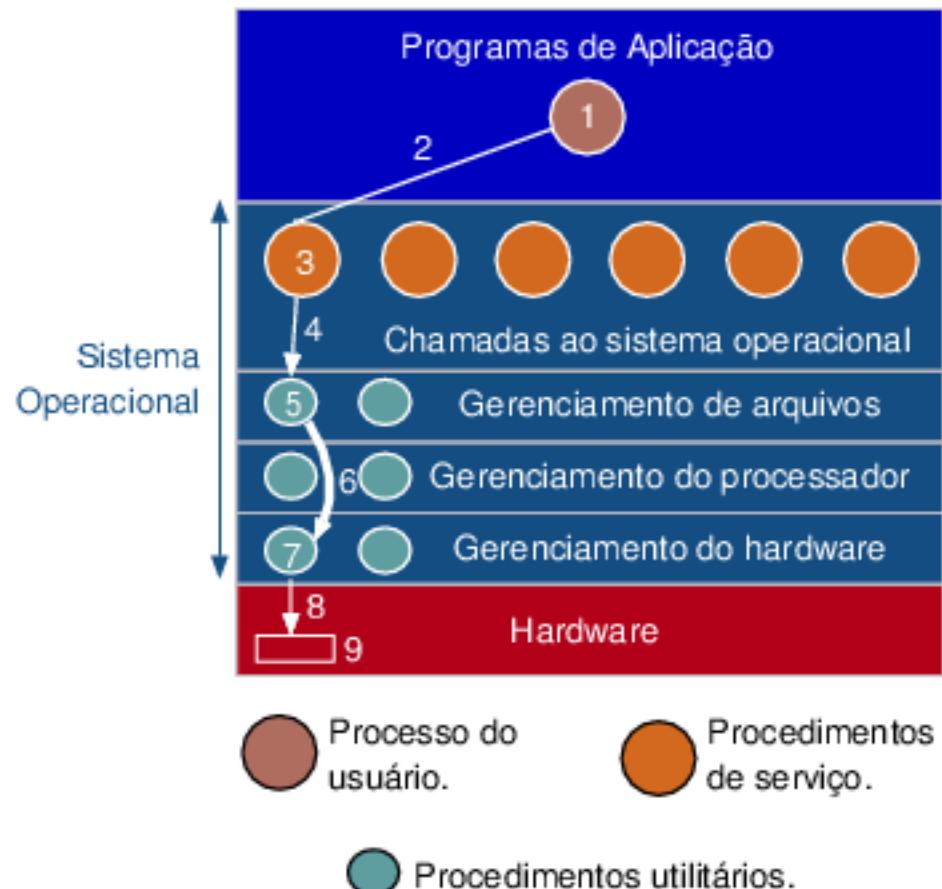


5: Um procedimento utilitário da camada de gerenciamento de arquivos começa a sua execução, e deverá então acessar a versão abstrata do disco real do computador, para ler as informações usadas para abrir o arquivo desejado.



Sistema em camadas

→ O sistema operacional é estruturado como uma hierarquia em camadas, onde cada camada implementa uma parte do sistema:

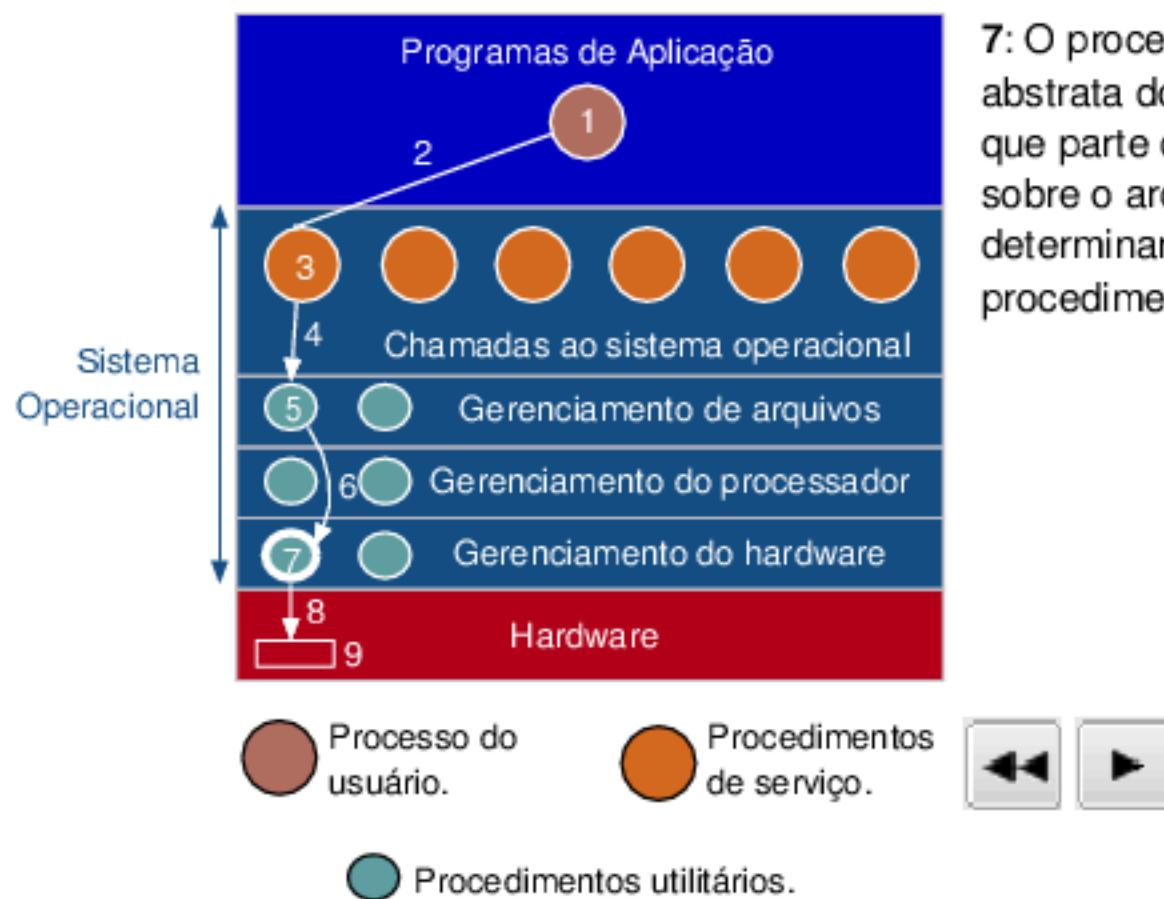


6: O uso do disco abstrato pelo procedimento da camada de gerenciamento de arquivos causa uma chamada a um procedimento utilitário da camada de gerenciamento do hardware, que implementa o disco abstrato.



Sistema em camadas

→ O sistema operacional é estruturado como uma hierarquia em camadas, onde cada camada implementa uma parte do sistema:

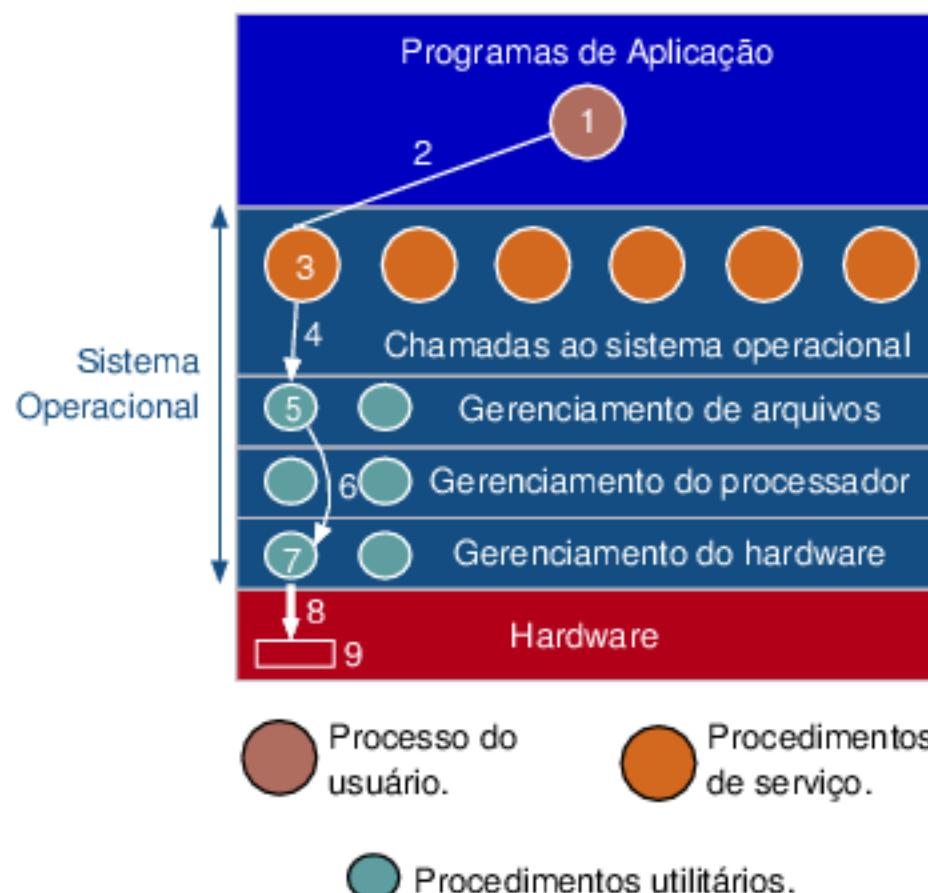


7: O procedimento que implementa a versão abstrata do disco é iniciado, e determina em que parte do disco estão as informações sobre o arquivo que deverá ser aberto. Após determinar onde estão estas informações, o procedimento iniciará um acesso ao disco.



Sistema em camadas

→ O sistema operacional é estruturado como uma hierarquia em camadas, onde cada camada implementa uma parte do sistema:



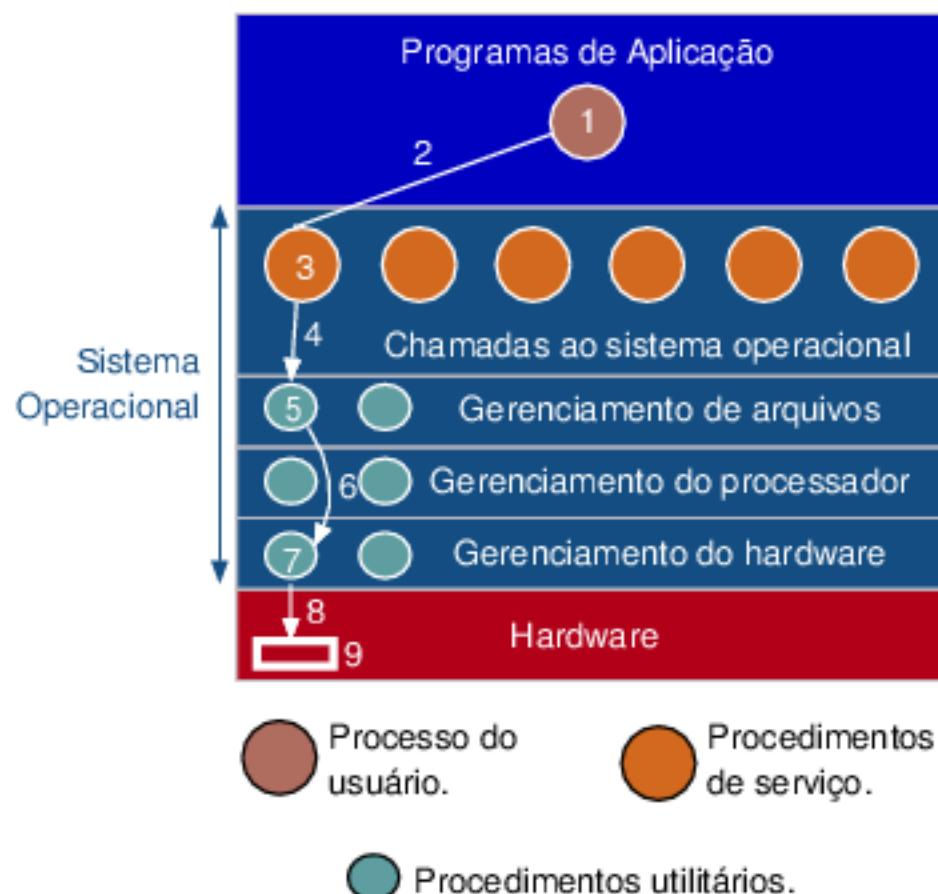
8: O procedimento que implementa o disco abstrato envia comandos ao dispositivo para iniciar uma operação de E/S que irá ler as informações do arquivo a ser aberto.



Procedimentos utilitários.

Sistema em camadas

→ O sistema operacional é estruturado como uma hierarquia em camadas, onde cada camada implementa uma parte do sistema:

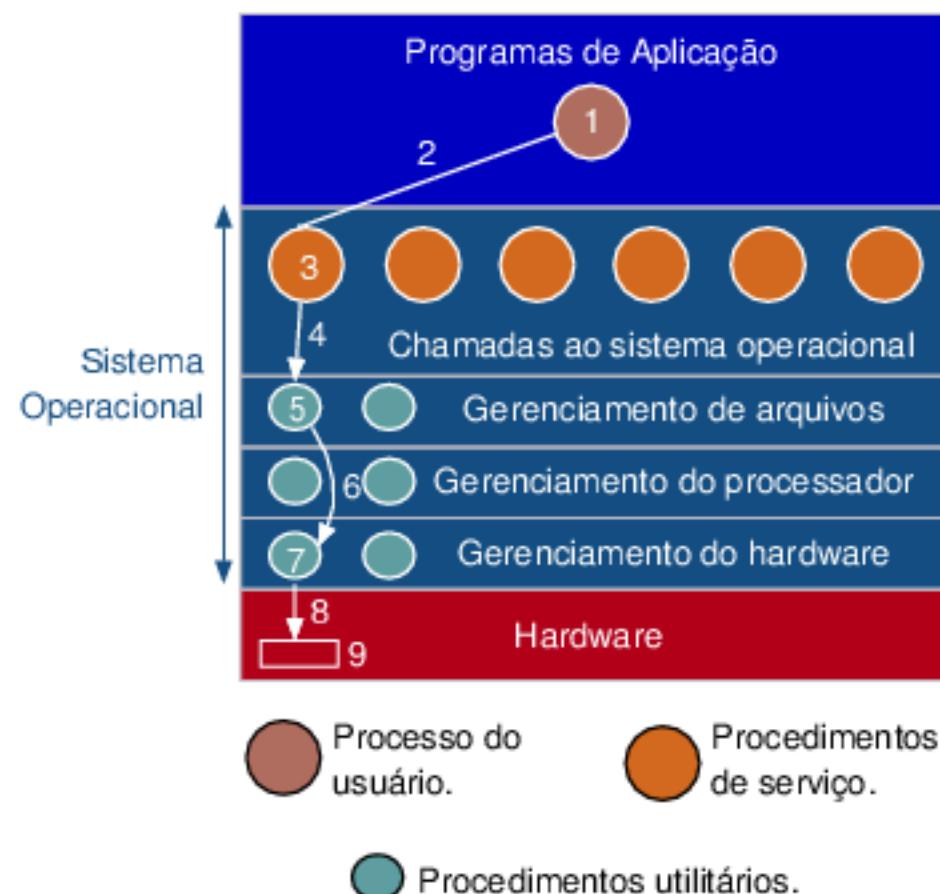


9: Os circuitos do disco rígido procuram as informações do arquivo a ser aberto. Neste caso, o processo será suspenso até que o disco leia todas as informações, terminando assim a operação de E/S iniciada.



Sistema em camadas

→ O sistema operacional é estruturado como uma hierarquia em camadas, onde cada camada implementa uma parte do sistema:

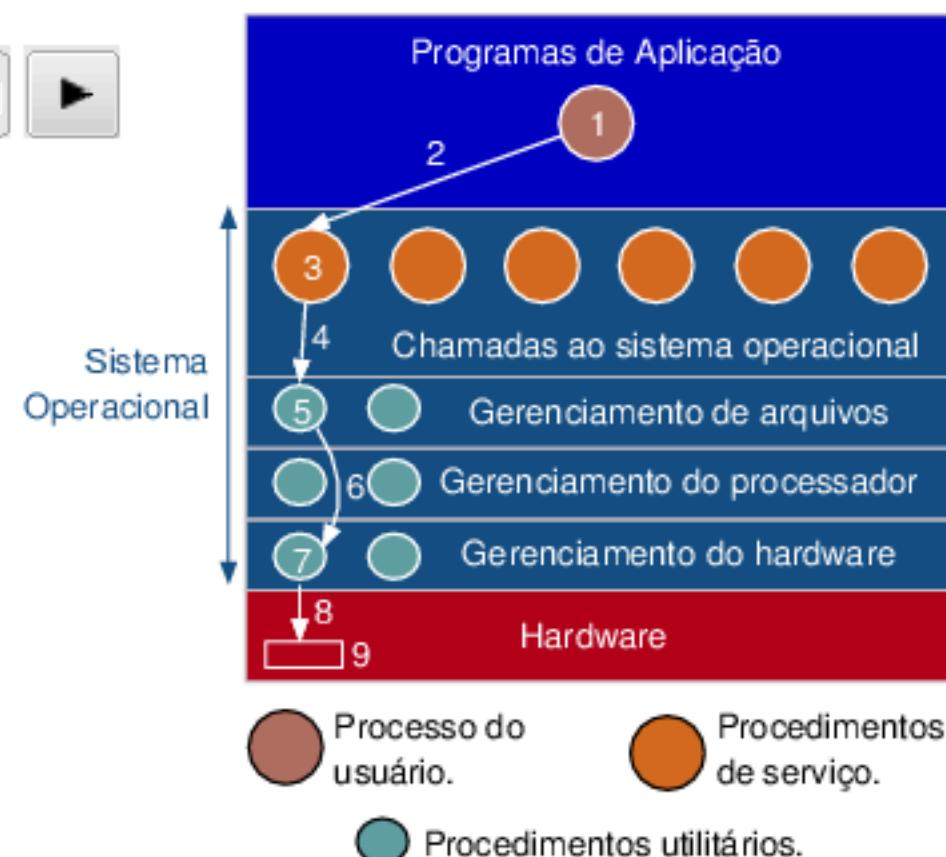


Depois do disco ler estas informações, o controle eventualmente é retornado ao procedimento que trata do disco abstrato, que depois retornará o controle ao procedimento que trata do gerenciamento do sistema de arquivos, que passará as informações do arquivo ao procedimento para abrir um arquivo, que então abrirá o arquivo. Finalmente, a biblioteca retornará um identificador para este arquivo ao processo do usuário, que continuará a sua execução.



Sistema em camadas

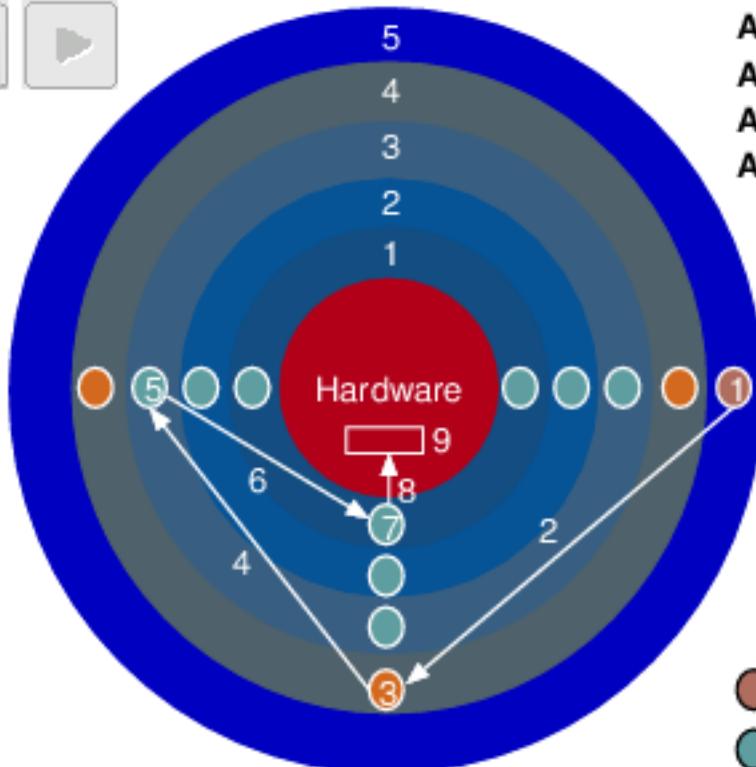
- ➡ O sistema MULTICS propôs uma estruturação alternativa, baseada em um conjunto de anéis concêntricos.
- ➡ Estruturação em anéis para o nosso exemplo anterior:



A hierarquia em camadas somente fornece uma estruturação para o núcleo do sistema operacional, mas um nível não é obrigado a usar os serviços de um nível inferior. No nosso exemplo, o procedimento utilitário da camada de gerenciamento de arquivos poderia usar o disco real do computador ao invés do abstrato.

Sistema em camadas

- O sistema MULTICS propôs uma estruturação alternativa, baseada em um conjunto de anéis concêntricos.
- Estruturação em anéis para o nosso exemplo anterior:



Anel 1: Gerenciamento do hardware.
Anel 2: Gerenciamento do processador.
Anel 3: Gerenciamento de arquivos.
Anel 4: Chamadas ao sistema operacional.
Anel 5: Programas do usuário.

Na hierarquia em anéis, os anéis mais externos somente podem acessar os mais internos de modo similar a como um processo do usuário faz as chamadas ao sistema operacional. Com isso, o hardware força que a abstração definida pelos anéis internos para os externos seja respeitada, o que não ocorre no sistema baseado em camadas.

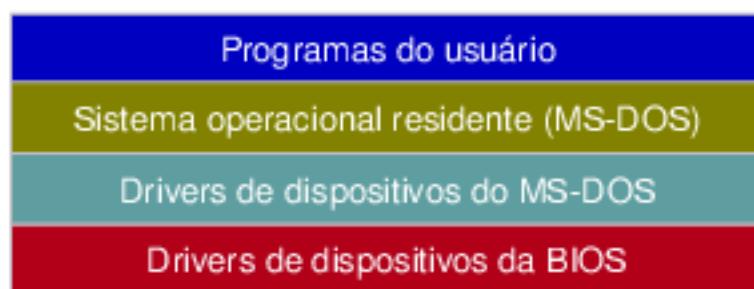
- Processo do usuário.
- Procedimentos utilitários.
- Procedimentos de serviço.

Sistema em camadas

- ➡ O sistema em camadas não impede que um procedimento em uma camada chame um procedimento de uma camada inferior:
 - No exemplo anterior, o procedimento utilitário poderia acessar diretamente o disco real ao invés de usar o disco abstrato.
- ➡ Um sistema em camadas somente define uma estruturação para o núcleo do sistema, e não uma proteção de acesso ao hardware.
- ➡ A estruturação baseada em anéis possui as seguintes vantagens sobre a estruturação em camadas:
 - Cada anel possui uma prioridade de acesso.
 - A prioridade do anel depende da posição do anel na hierarquia.
 - A proteção de acesso é assegurada pelo hardware.

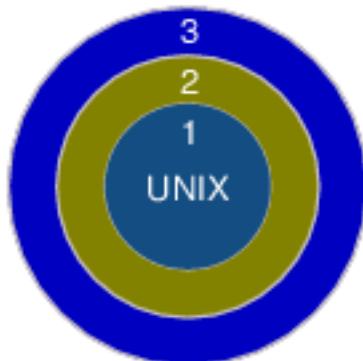
Sistema em camadas

- ▶ Um exemplo tradicional de um sistema em camadas é o THE, criado por Dijkstra em 1968:
- ▶ Um outro exemplo de um sistema em camadas é o MS-DOS:



Pressione o mouse sobre um dos níveis para ver o resumo da função deste nível.

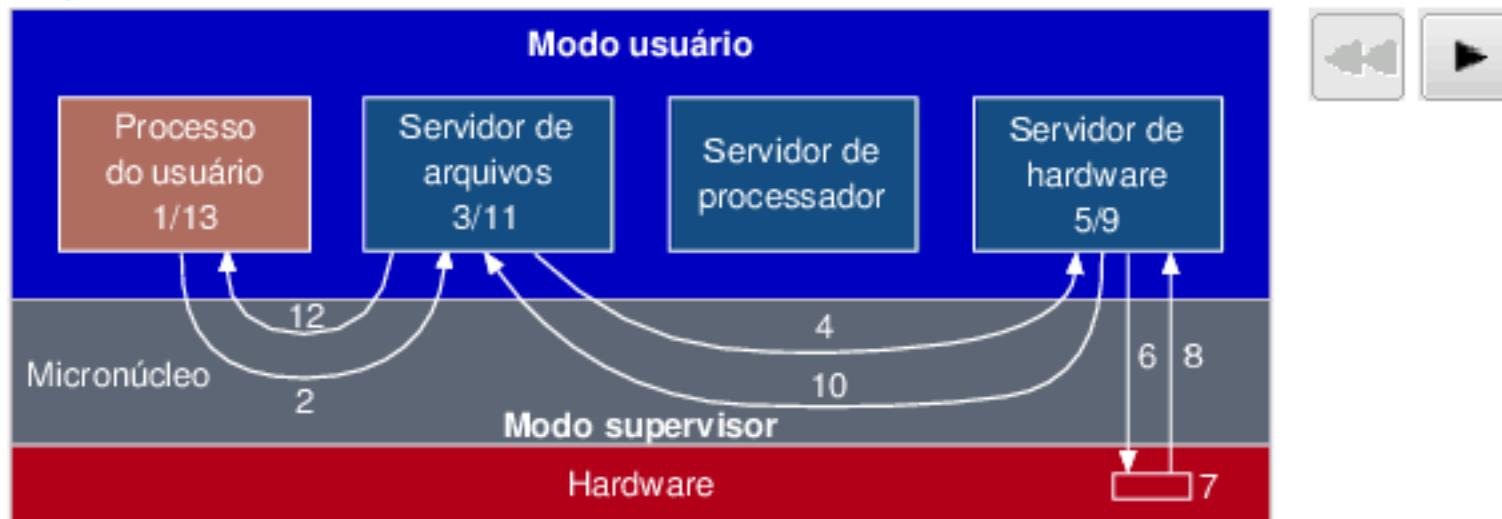
- ▶ O UNIX é um exemplo de um sistema baseado em anéis:



Pressione o mouse sobre um dos anéis para ver o resumo da função deste anel.

Modelo cliente-servidor

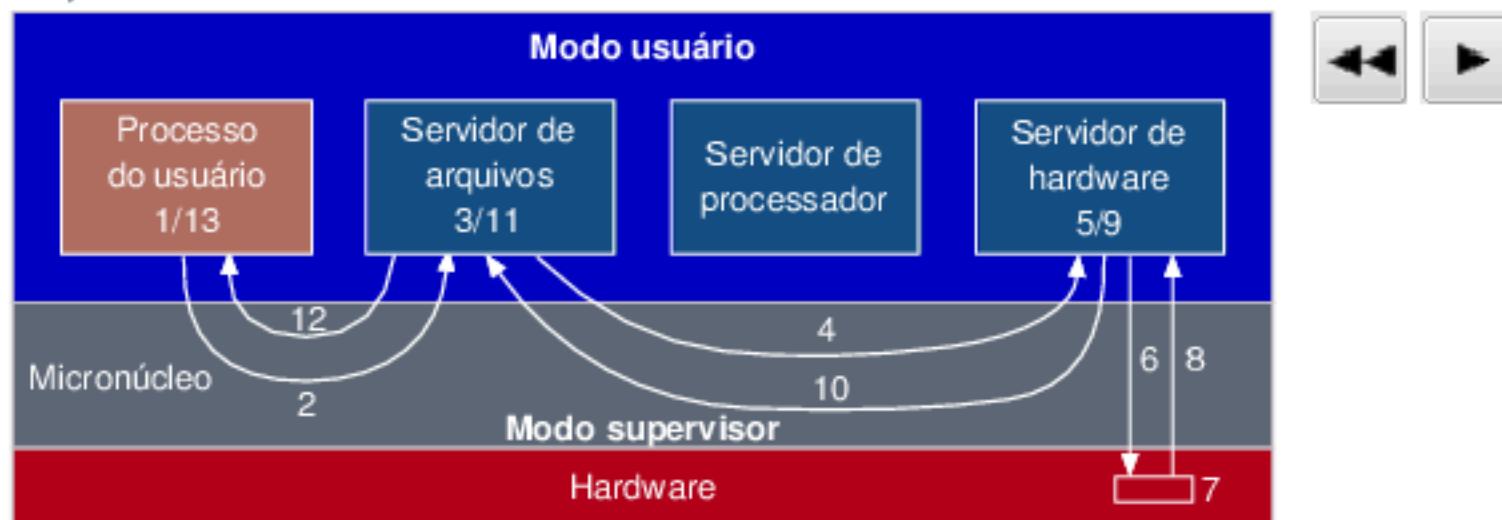
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



Na estruturação do sistema segundo o modelo cliente-servidor, o núcleo do sistema, que é chamado de **micronúcleo**, trata somente da troca de mensagens entre os processos rodando no modo usuário, e do acesso aos dispositivos físicos, mas não do seu gerenciamento. Os outras partes do sistema são processos que executam no modo usuário.

Modelo cliente-servidor

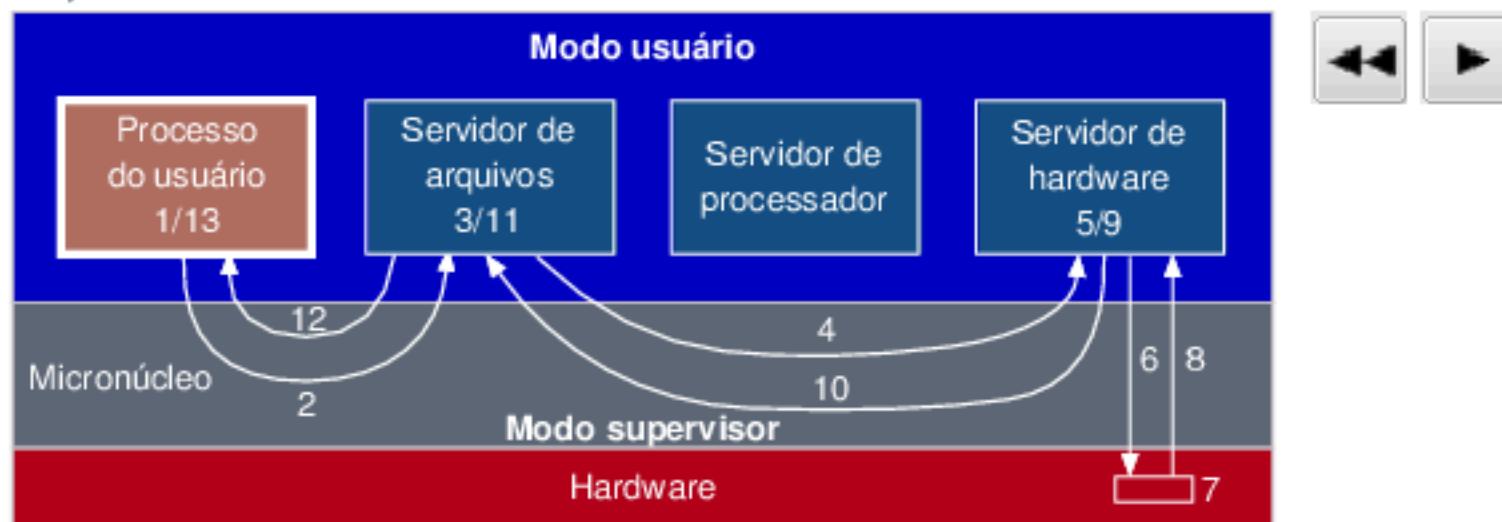
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



O processo agora deve enviar uma mensagem ao servidor, ao invés de usar a chamada ao sistema. O nosso exemplo pode usar três servidores: um de **arquivos**, que gerencia o sistema de arquivos, um de **processos**, que gerencia a alocação do processador, e um de **hardware**, que gerencia os dispositivos físicos do hardware.

Modelo cliente-servidor

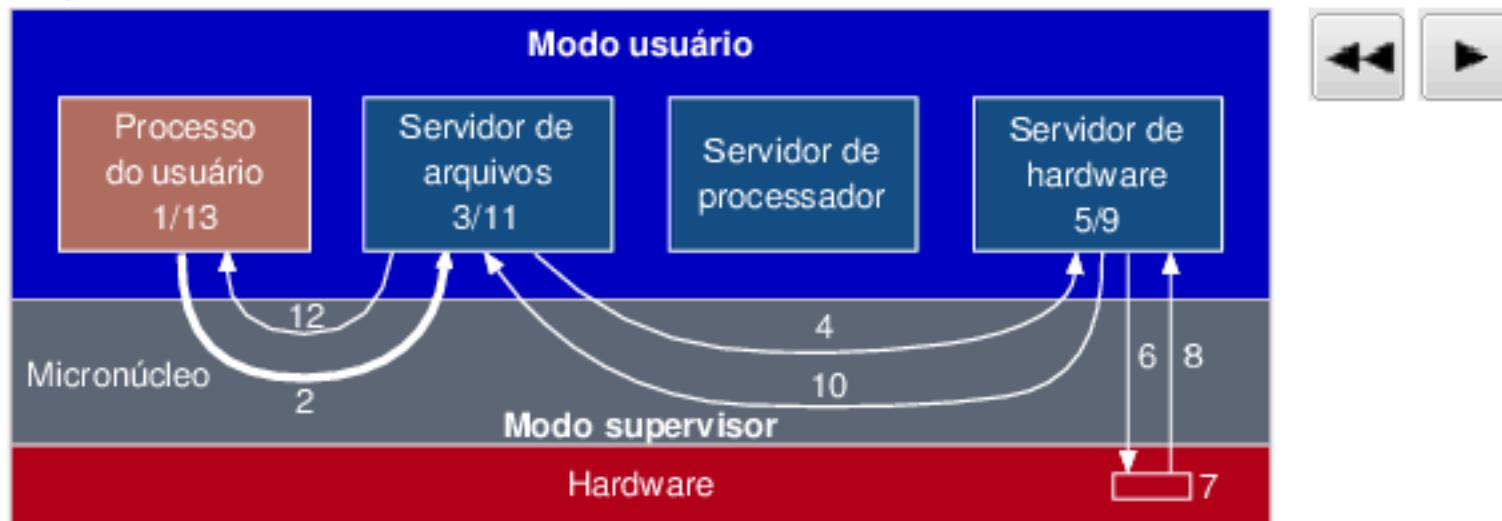
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



1: O processo do usuário estava executando e agora precisa abrir um arquivo no disco para poder continuar a sua execução. Para poder abrir este arquivo, o processo deverá enviar uma mensagem ao servidor de arquivos.

Modelo cliente-servidor

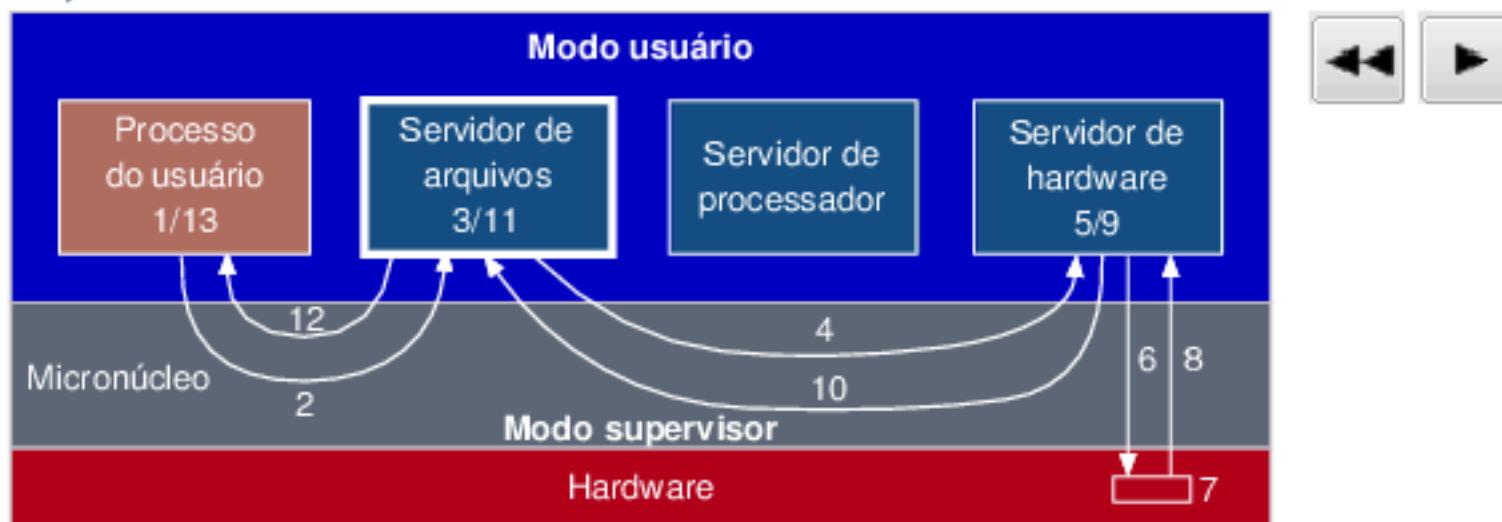
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



2: O processo do usuário então envia uma mensagem ao servidor de arquivos. A mensagem é enviada ao processo servidor pelo micronúcleo. O gerenciamento da troca de mensagens entre clientes e servidores é uma das poucas funções que o micronúcleo executa.

Modelo cliente-servidor

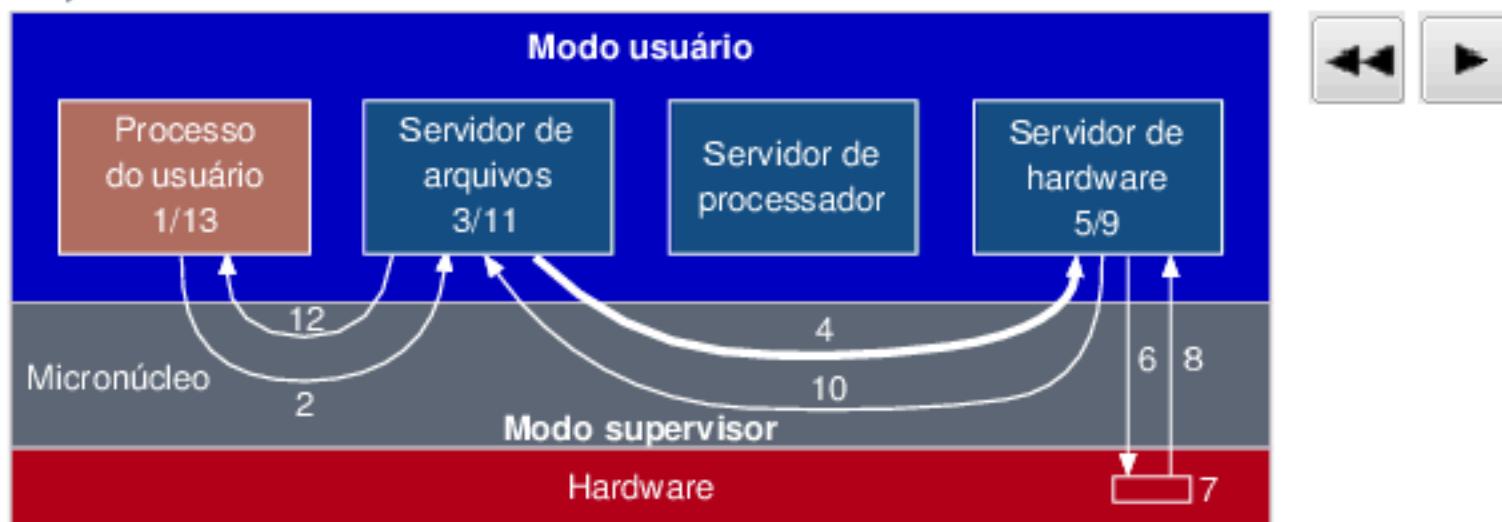
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



3: O processo servidor de arquivos, que gerencia o sistema de arquivos do disco, é iniciado. Para poder abrir o arquivo, o processo servidor deverá ler, assim como antes, as informações do arquivo no sistema de arquivos armazenado no disco abstrato. Com isso, o servidor então enviará uma mensagem ao servidor de hardware.

Modelo cliente-servidor

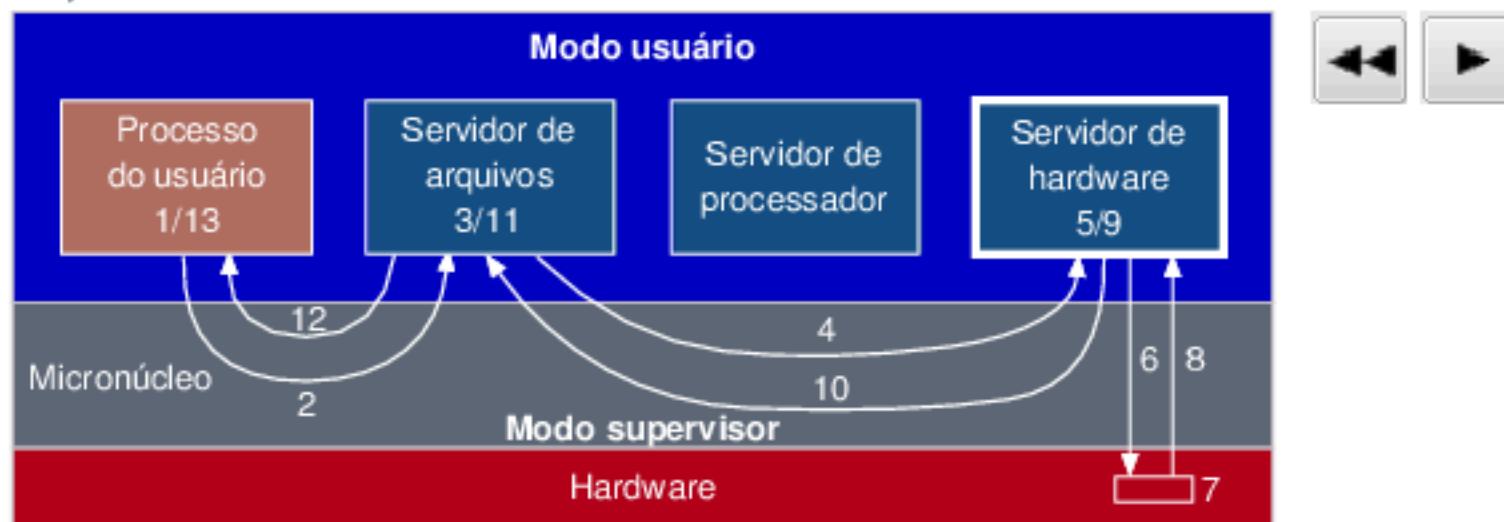
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



4: O servidor de arquivos envia então uma mensagem ao servidor de hardware, para que este acesse o disco abstrato.

Modelo cliente-servidor

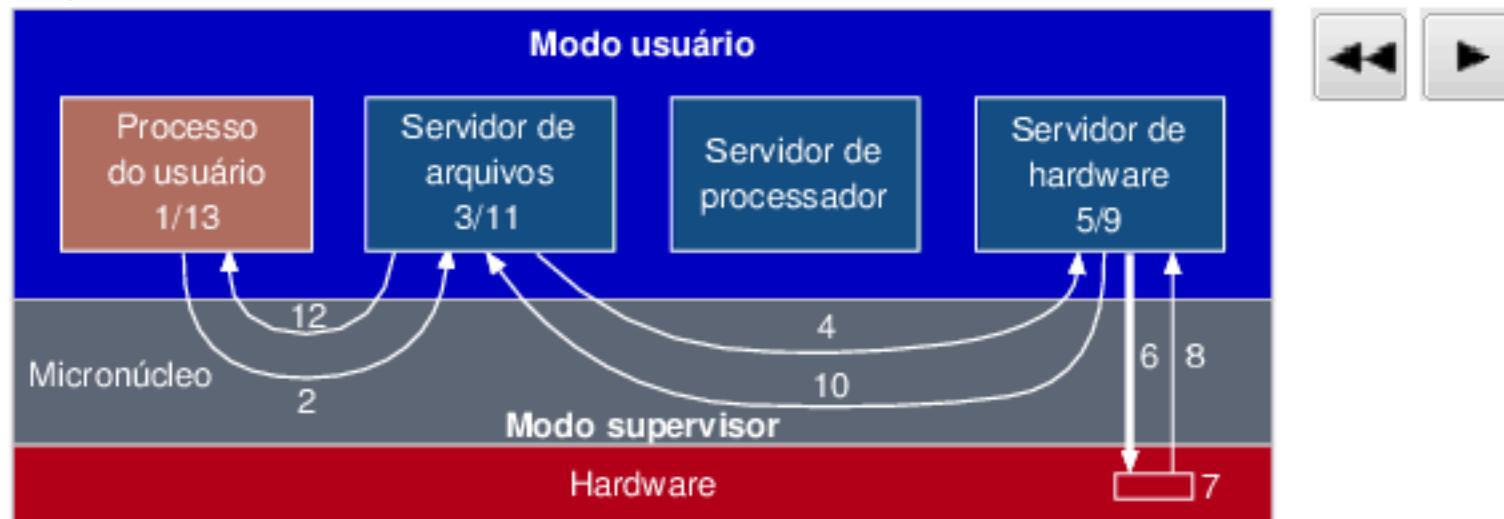
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



5: O servidor de hardware começa a sua execução, e descobre que um processo deseja acessar o disco abstrato. Para poder gerenciar este disco, o servidor deverá acessar o disco real do computador. Então, o servidor criará uma mensagem especial, informando ao micronúcleo que desejamos acessar um dos dispositivos físicos, no caso, o disco rígido.

Modelo cliente-servidor

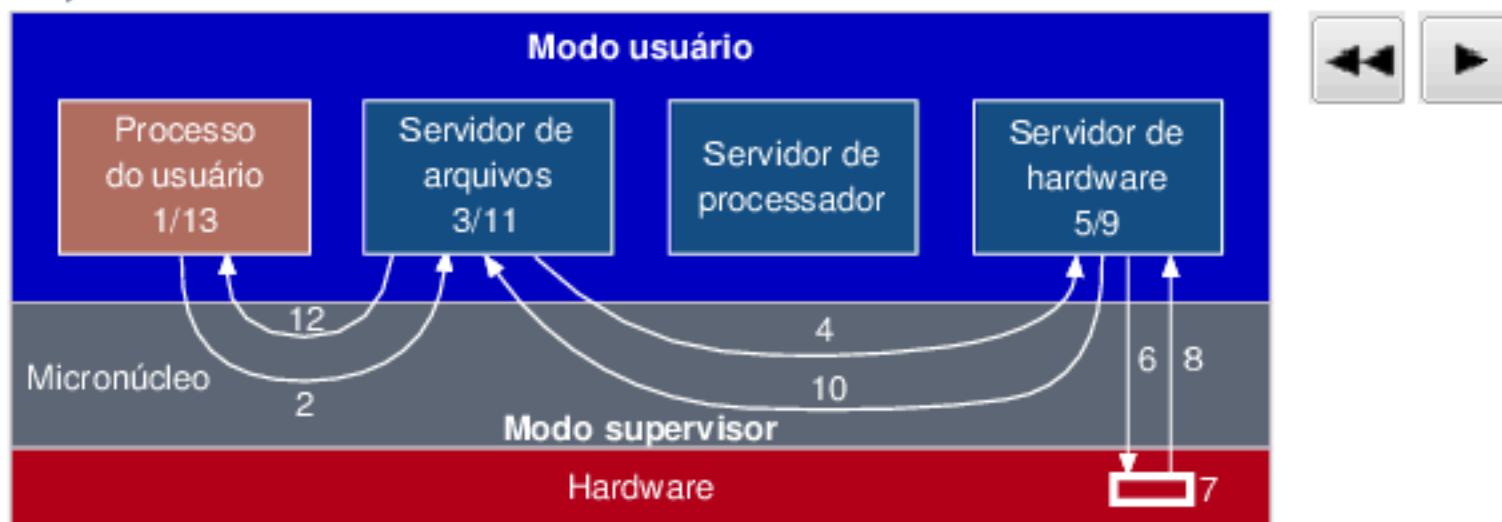
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



6: A mensagem especial, que informa que desejamos ler um conjunto de dados do disco rígido real do computador, é enviada pelo servidor de hardware. O micronúcleo detecta que a mensagem é especial, e usa as informações contidas nesta mensagem para informar ao controlador do disco rígido que desejamos ler uma região do disco.

Modelo cliente-servidor

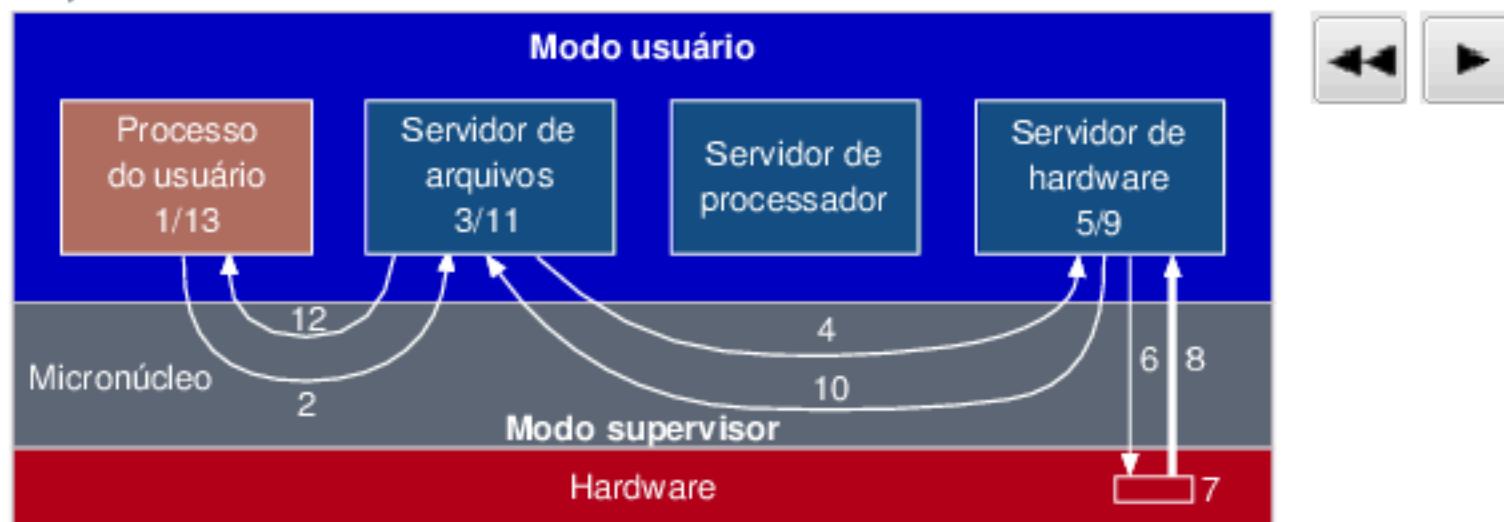
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



7: Os circuitos do disco rígido procuram pela região do disco, e depois disso, começam a ler os dados da região do disco que o servidor de hardware desejava acessar.

Modelo cliente-servidor

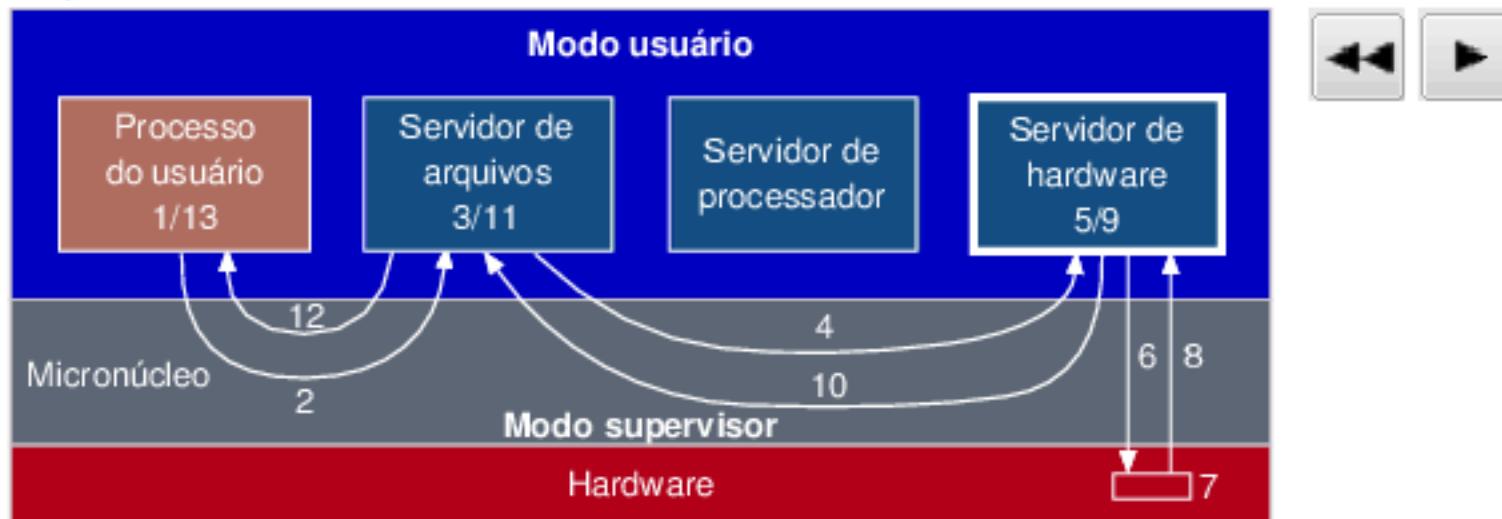
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



8: Depois de o disco rígido ler os dados solicitados pelo servidor de hardware, o micronúcleo envia uma mensagem a este servidor informando que os dados do disco já estão disponíveis para serem usados, em, por exemplo, uma dada região da memória do computador.

Modelo cliente-servidor

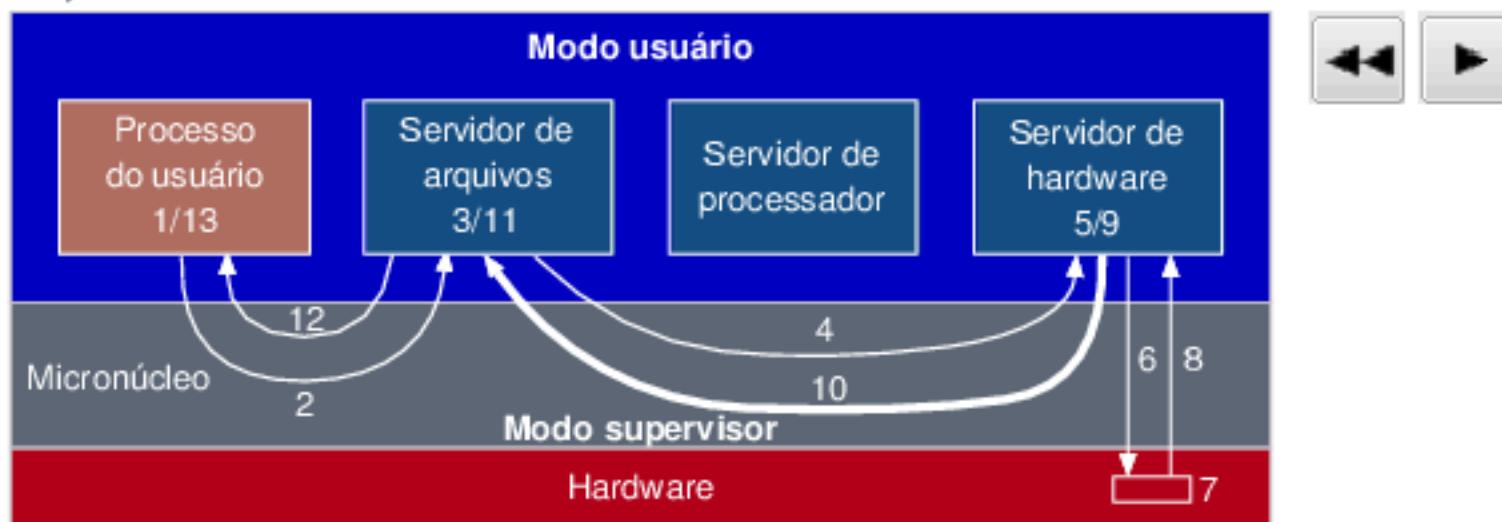
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



9: Depois de os dados serem lidos do disco rígido, o servidor de hardware pode executar algumas tarefas adicionais necessárias ao gerenciamento do disco abstrato, antes de finalmente enviar uma mensagem ao servidor de arquivos informando que a região do disco abstrato já foi lida, e que os dados já estão disponíveis para serem usados.

Modelo cliente-servidor

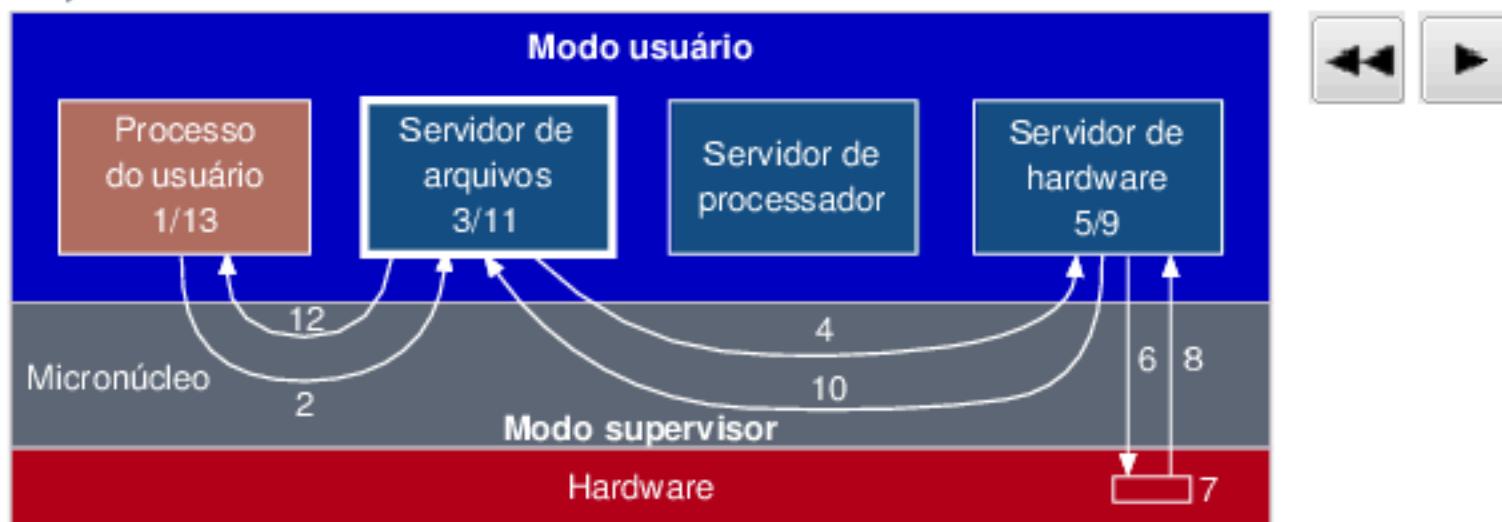
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



10: Uma mensagem então é enviada do servidor de hardware para o servidor de arquivos, informando que os dados lidos do disco abstrato já estão disponíveis para serem usados.

Modelo cliente-servidor

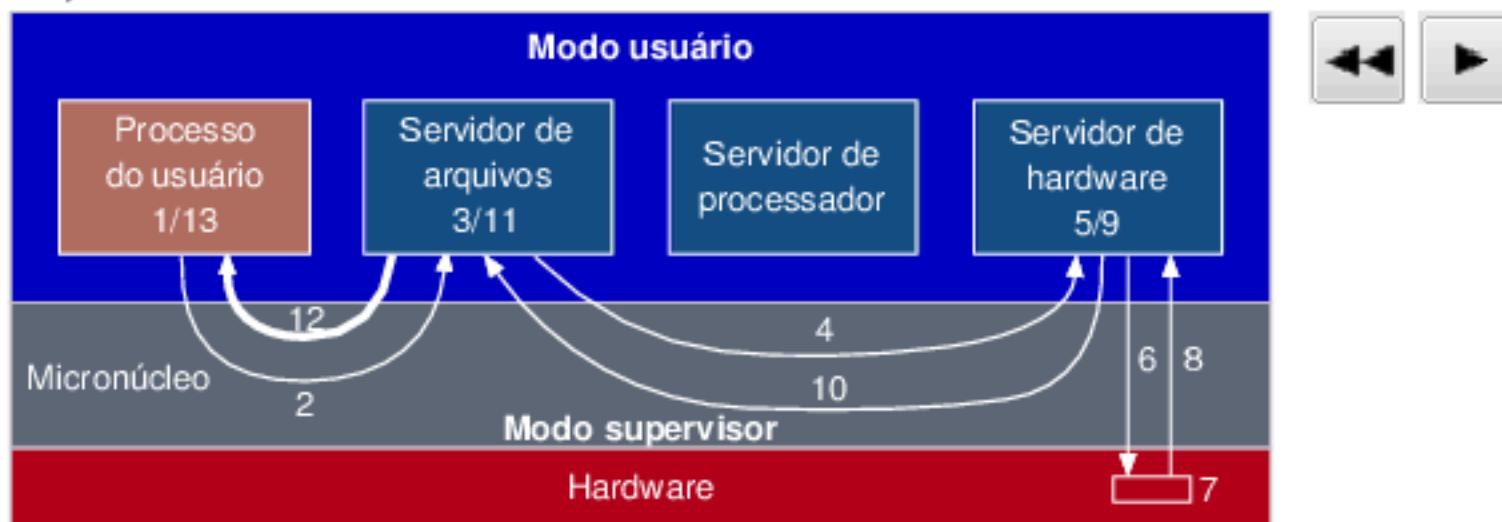
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



11: O servidor de arquivos usa os dados lidos do disco abstrato para poder obter as informações necessárias para abrir o arquivo requisitado pelo processo do usuário.

Modelo cliente-servidor

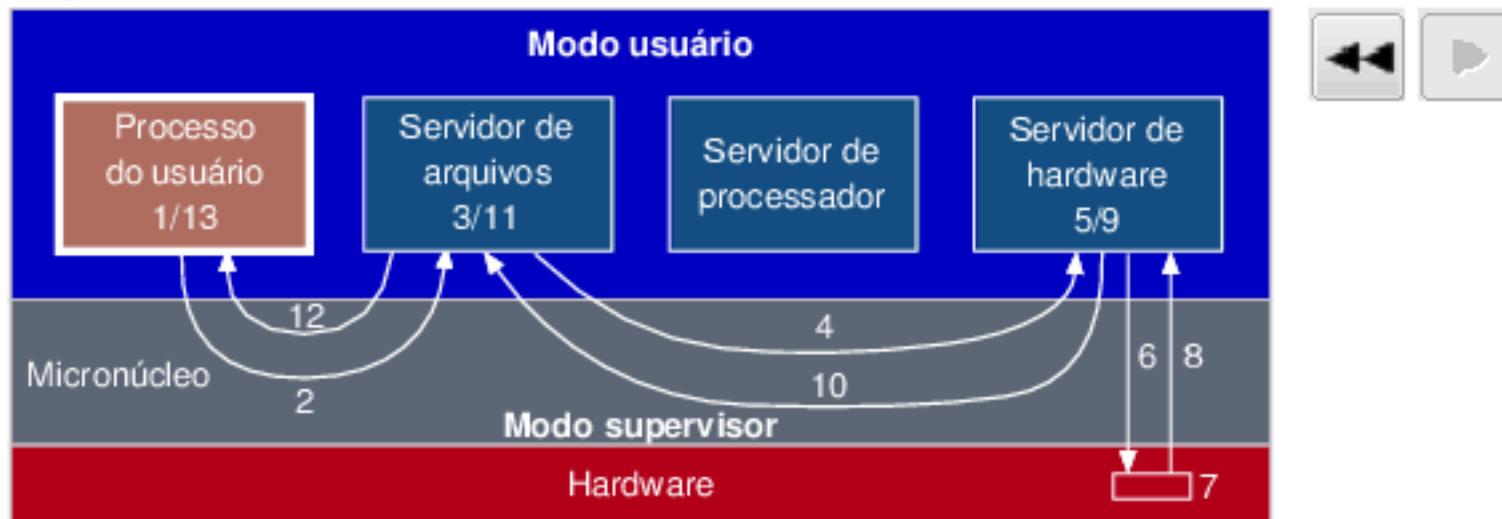
- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



12: O servidor de arquivos envia uma mensagem para o processo do usuário, com o identificador do arquivo que o processo do usuário desejava acessar.

Modelo cliente-servidor

- ▶ Baseado no conceito de clientes e de servidores.
- ▶ O núcleo do sistema, chamado de **micronúcleo**, essencialmente trata da troca de mensagens entre clientes e servidores.
- ▶ Possível estruturação para o nosso exemplo anterior:



13: O processo do usuário, que agora pode continuar a sua execução, usa o identificador do arquivo, contido na mensagem enviada pelo servidor de arquivos, para poder acessar o arquivo.

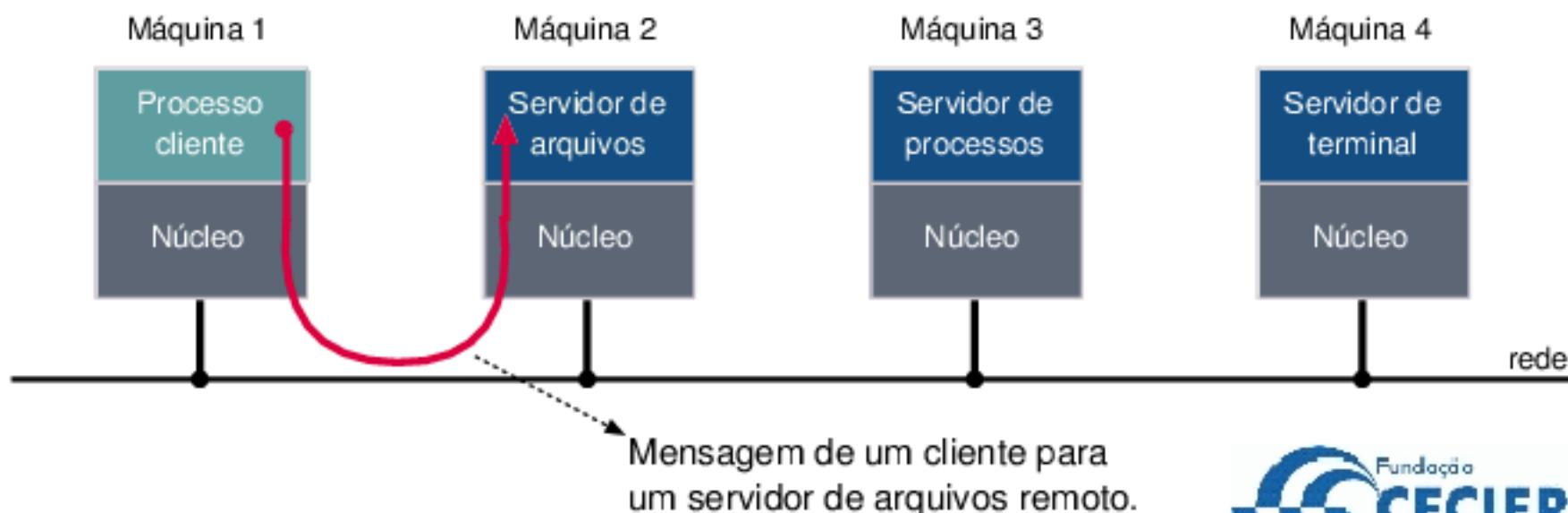
Modelo cliente-servidor

- ➡ Objetiva reduzir o código do núcleo do sistema, colocando quase todas as partes do sistema executando no modo usuário.
- ➡ Baseado no modelo cliente-servidor:
 - Existem processos servidores, rodando no modo usuário, executando quase todos os serviços do sistema.
 - O processo cliente que deseja um serviço do sistema envia uma mensagem ao processo servidor que executa o serviço.
 - Ao receber uma mensagem de um cliente, o processo servidor executa o serviço, e depois envia a resposta a este cliente.
- ➡ O micronúcleo trata da troca de mensagens entre clientes e servidores, e somente do acesso aos dispositivos físicos.

Modelo cliente-servidor

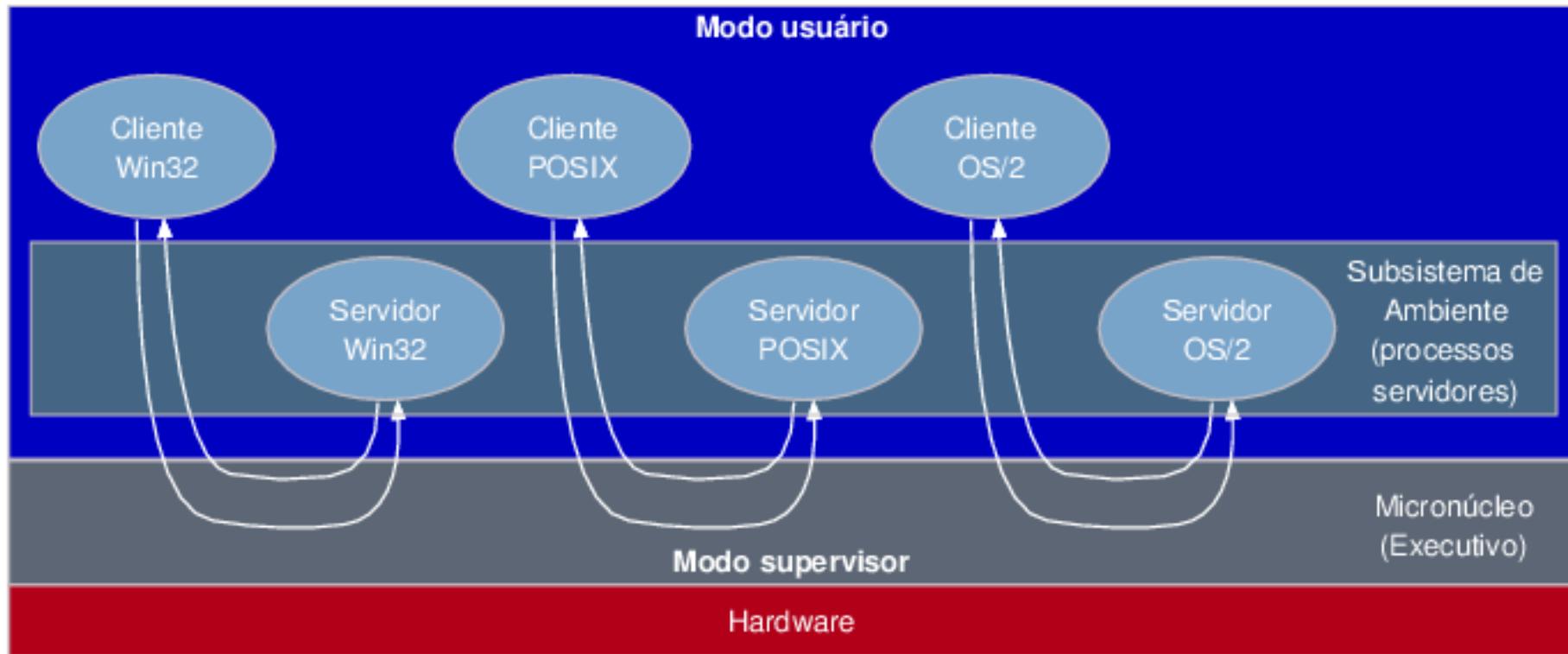
→ Vantagens do modelo cliente-servidor:

- O sistema é dividido em partes menores que são mais fáceis de manter e de gerenciar.
- Como os servidores executam no modo usuário, um erro de programação em um deles não compromete o sistema.
- Facilita a implementação de sistemas distribuídos, pois os servidores não precisam estar na mesma máquina.



Modelo cliente-servidor

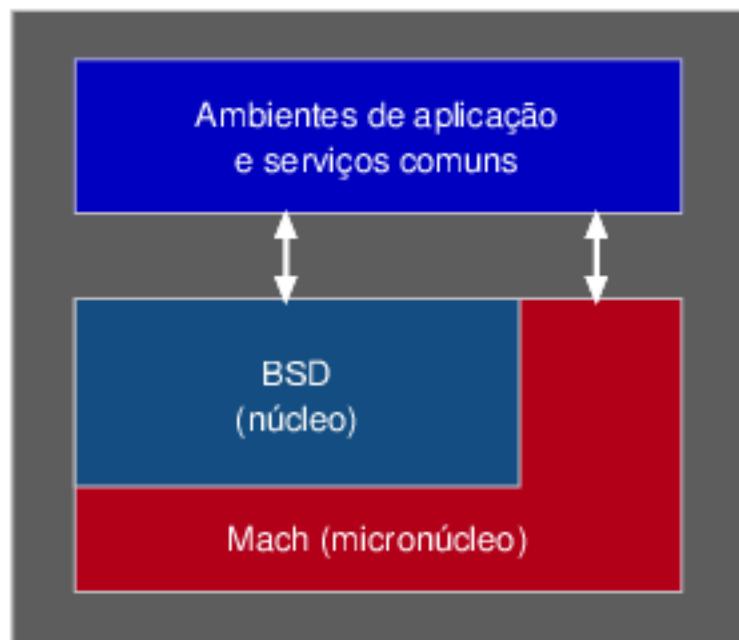
→ O Windows NT é um exemplo de sistema baseado neste modelo:



O micronúcleo do Windows NT, chamado de **Executivo**, gerencia a multiprogramação, a alocação da memória entre os processos, os objetos do sistema associados à implementação do sistema (como arquivos, janelas, e mensagens), o tratamento das interrupções, e o gerenciamento dos dispositivos de E/S.

Modelo cliente-servidor

- O Mac OS X é um outro exemplo de sistema baseado no modelo:



O sistema MAC OS X, que executa nos computadores da linha Machintosh, é baseado no sistema NeXTStep, que usa o micronúcleo Mach para gerenciar o hardware do computador. Existe um único servidor rodando no sistema, que é uma versão do núcleo do sistema BSD (baseado no sistema UNIX). O BSD acessa o hardware da máquina através do Mach, e não diretamente através de drivers de dispositivo. Os programas ou usam as chamadas ao BSD, ou as chamadas ao Mach.

- O sistema operacional MINIX, a partir da versão 2.0, também é um exemplo de um sistema baseado no modelo cliente-servidor.