Multiplexação de E/S: Função `select`

MC 833 – Programação em Redes de Computadores
Instituto de Computação – UNICAMP

Juliana Freitag Borin

Conteúdo do capítulo 6 do livro da bibliografia principal.

Introdução

- Suponha uma aplicação cliente/servidor TCP de *echo* - o cliente envia uma mensagem para o servidor e o servidor faz um *echo* desta mensagem para o cliente.
  - o cliente está manipulando duas entradas simultaneamente: entrada padrão + socket TCP
  - essas entradas são bloqueantes

- Suponha ainda que o processo servidor que atende este cliente é "morto".

- O que acontece com o cliente, caso ele esteja bloqueado em uma chamada (`fgets`, p.ex.) para a entrada padrão?

Introdução

- Suponha uma aplicação cliente/servidor TCP de *echo* - o cliente envia uma mensagem para o servidor e o servidor faz um *echo* desta mensagem para o cliente.
  - o cliente está manipulando duas entradas simultaneamente: entrada padrão + socket TCP
  - essas entradas são bloqueantes

- Suponha ainda que o processo servidor que atende este cliente é "morto".

- O que acontece com o cliente, caso ele esteja bloqueado em uma chamada (`fgets`, p.ex.) para a entrada padrão?
  - Receberá um FIN do servidor, mas só verá a notificação quando voltar a ler do socket!

Multiplexação de E/S

- Multiplexação de E/S: permite avisar o kernel de que queremos ser notificados assim que condições de E/S estejam válidas. Ex.: dados para leitura estão disponíveis.

- `select` e `poll`

- Não só para sockets

Multiplexação de E/S - quando usar?

- Quando cliente está manipulando vários descritores. Ex.: descritor do socket é de entrada interativa.
- Quando o cliente manipula vários sockets ao mesmo tempo.
- Quando TCP manipula listening sockets e outros sockets conectados. Ex.: servidor concorrente sem `fork`.
- Quando o servidor lida com TCP e com UDP simultaneamente.
- Quando o servidor manipula vários protocolos e serviços simultaneamente. Ex.: inetd.

E/S - fases e modelos

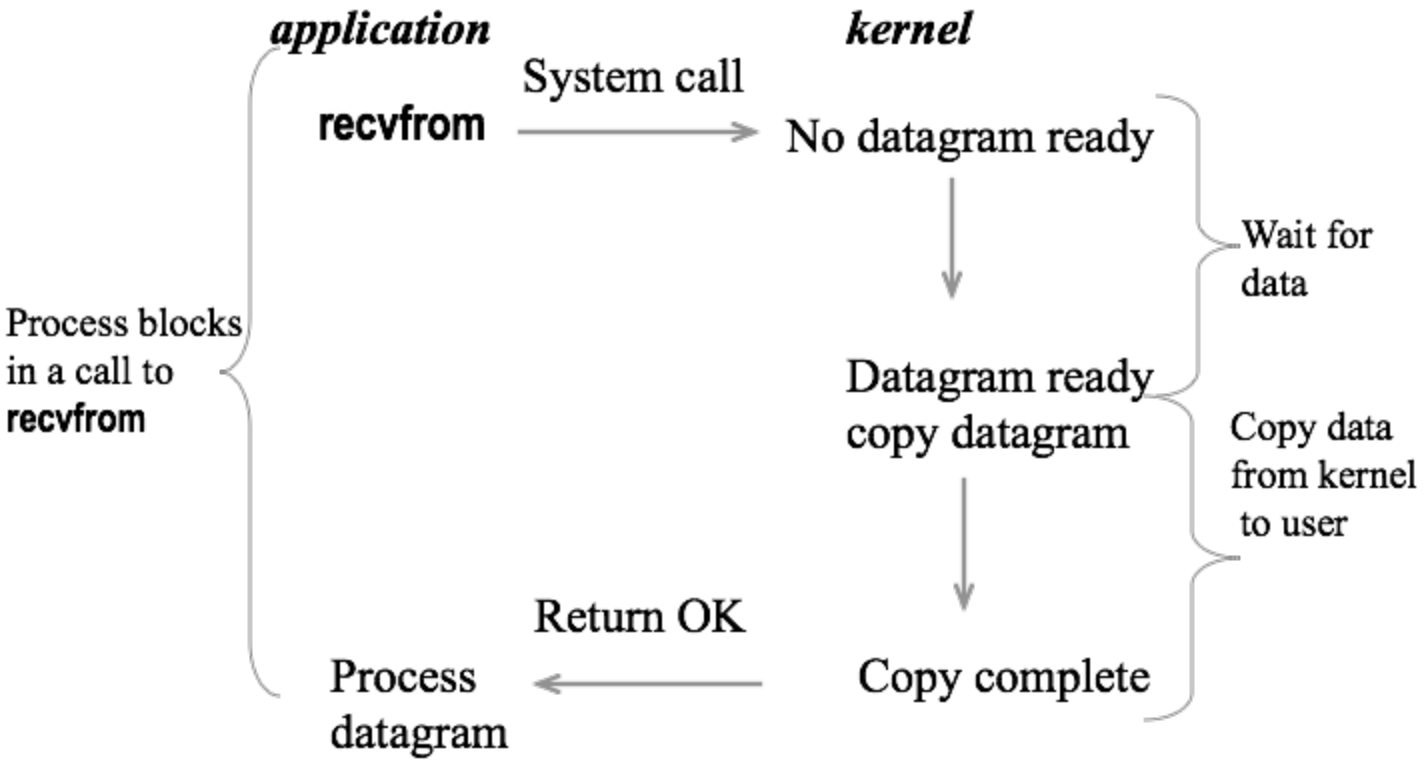
Operações de entrada são normalmente compostas por 2 fases:

- espera até os dados estarem disponíves;
  - socket: espera a chegada dos dados pela rede e, em seguida, copia os dados para um buffer do kernel.
- cópia dos dados do kernel para o processo.
  - socket: copia os dados do buffer do kernel para o buffer da aplicação.

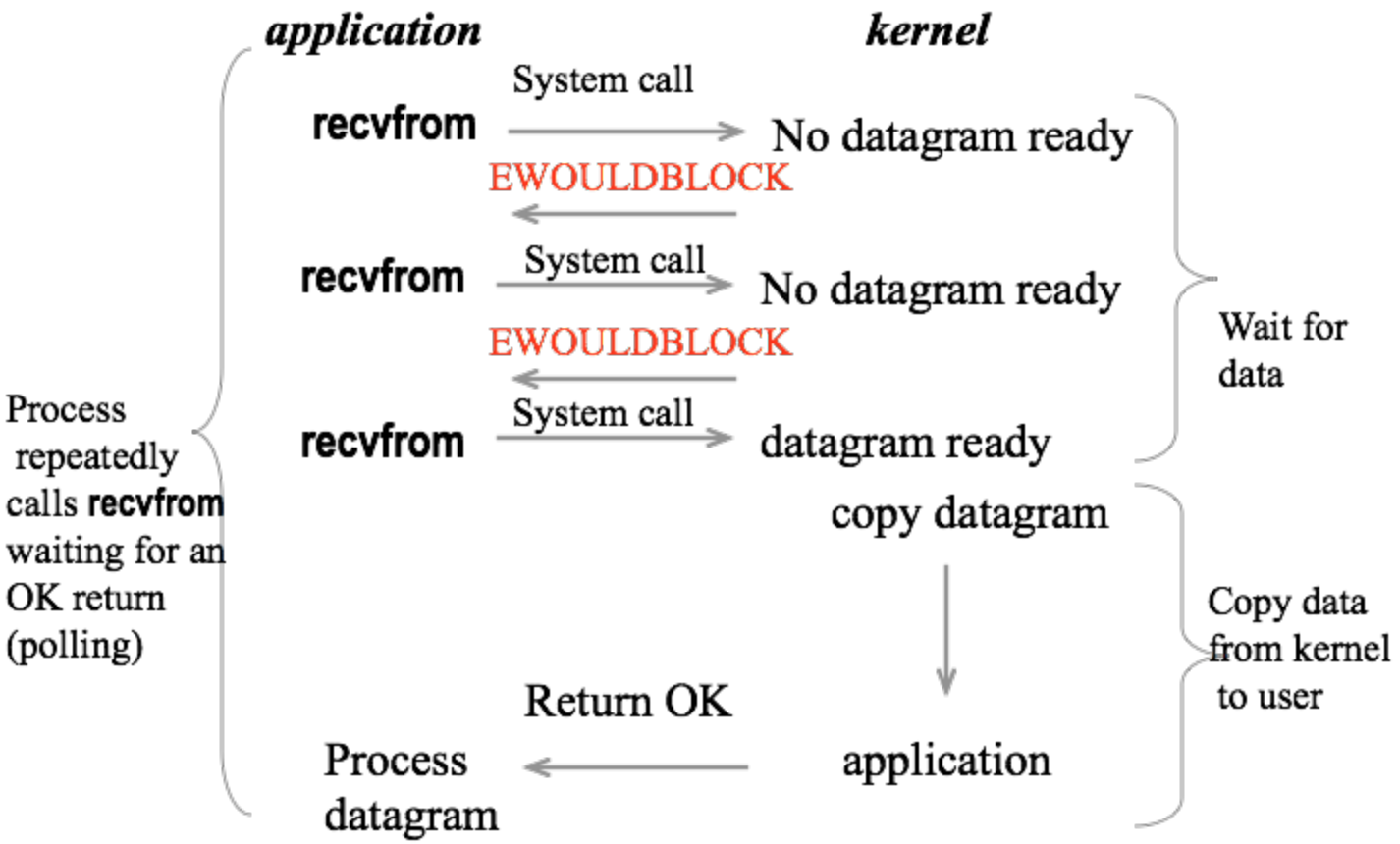
Modelos de E/S:

- E/S bloqueante
- E/S não bloqueante
- Multiplexação de E/S
- E/S orientada a sinal
- E/S assíncrona

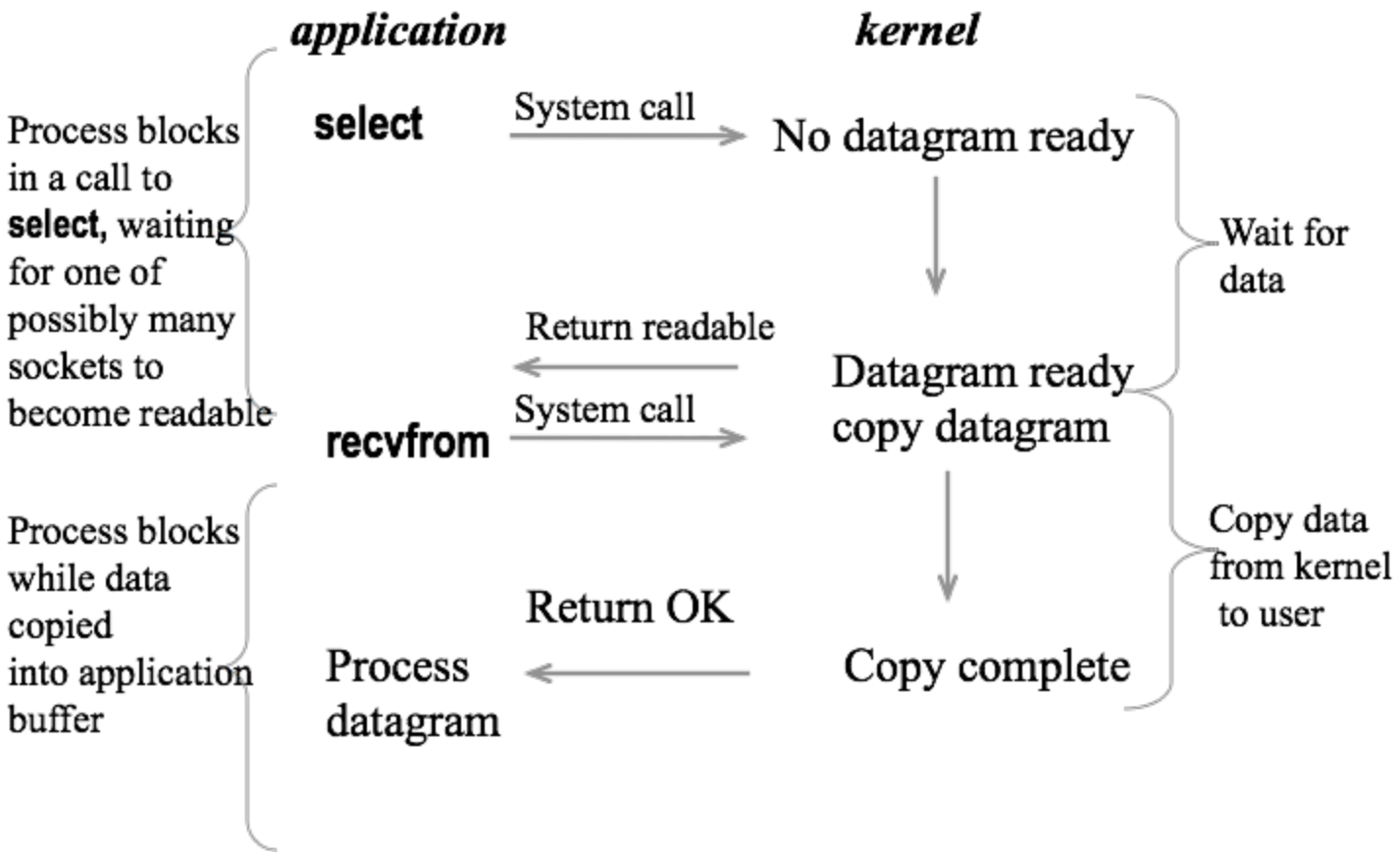
E/S bloqueante



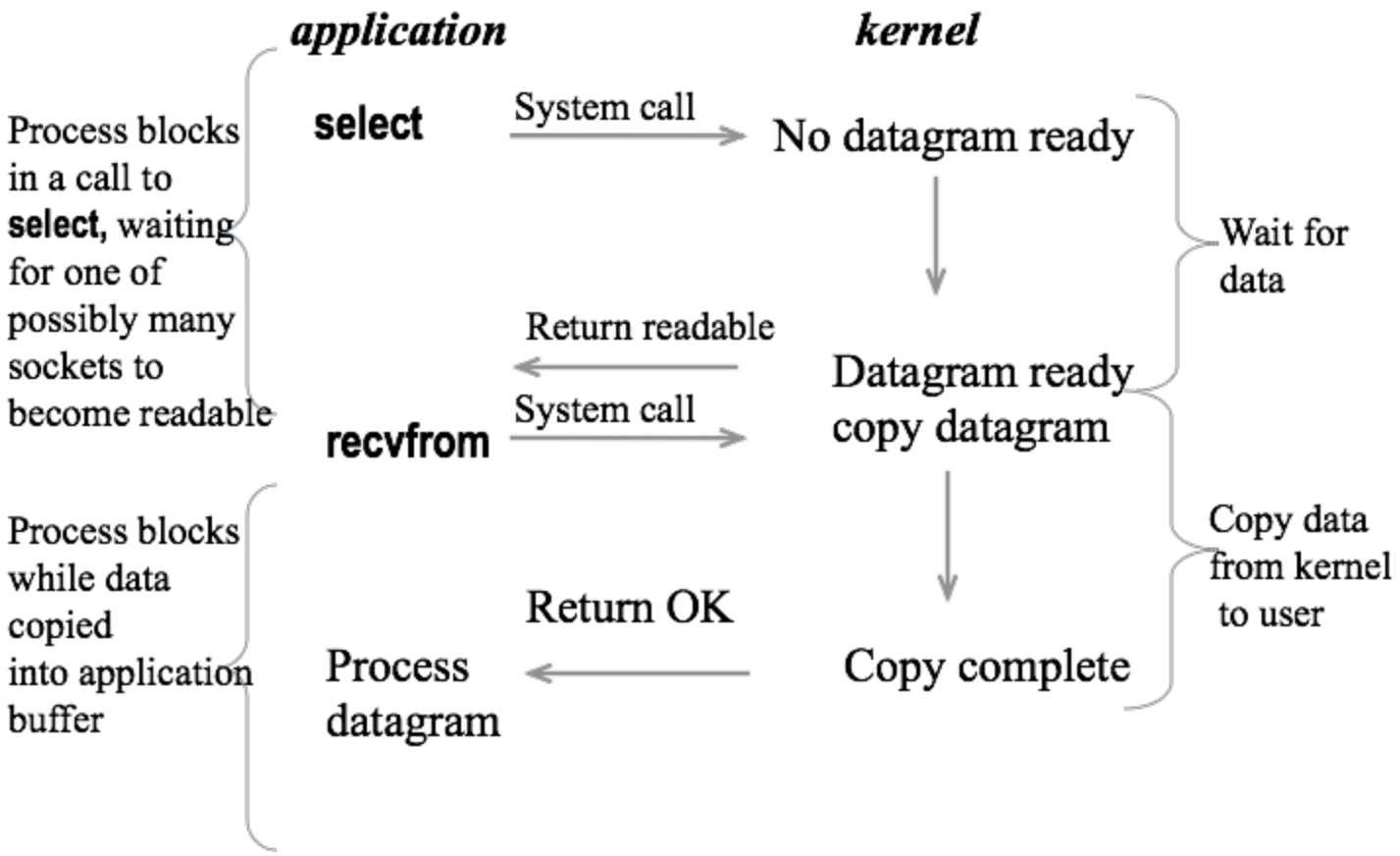
E/S não bloqueante



Multiplexação de E/S

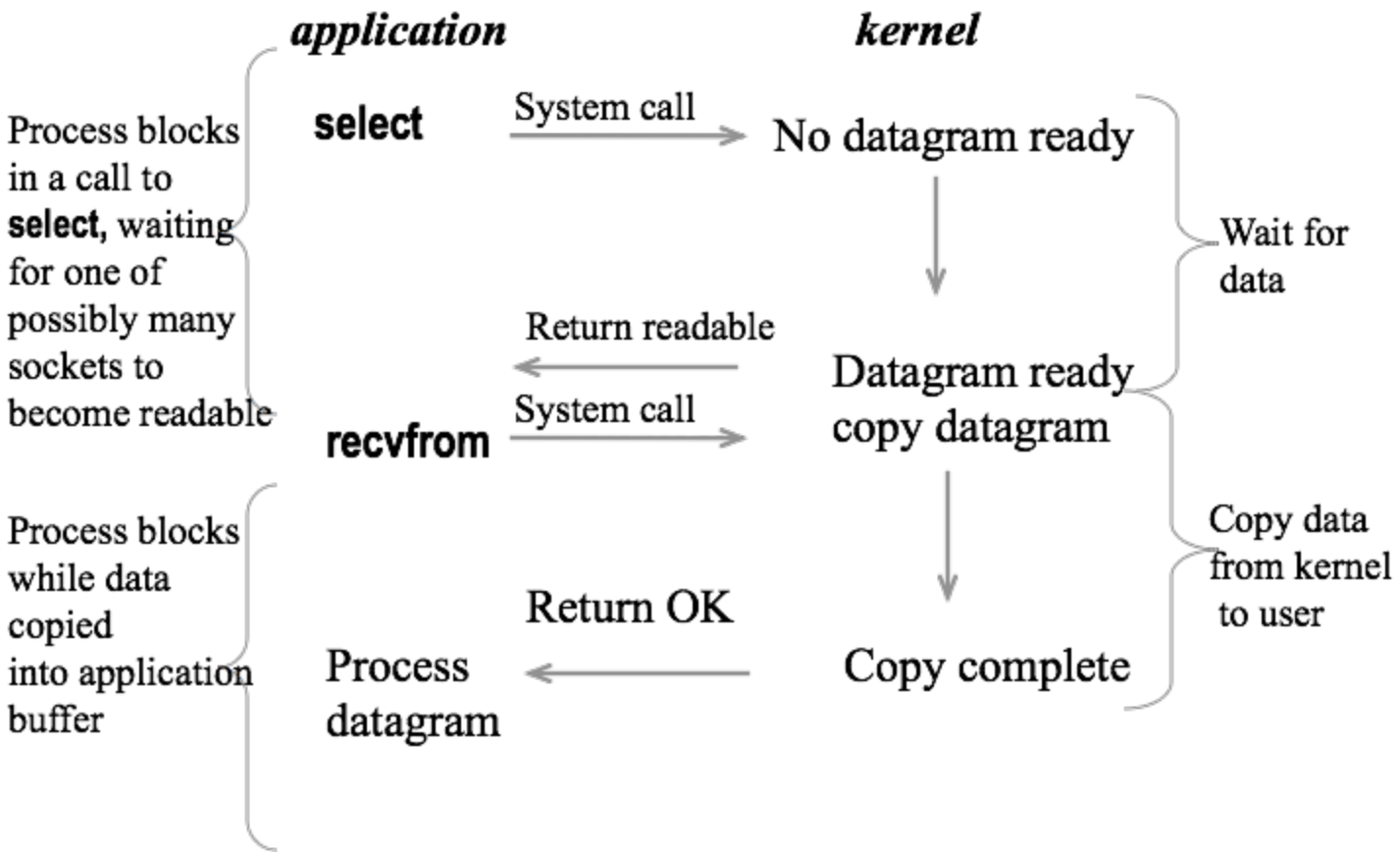


Multiplexação de E/S



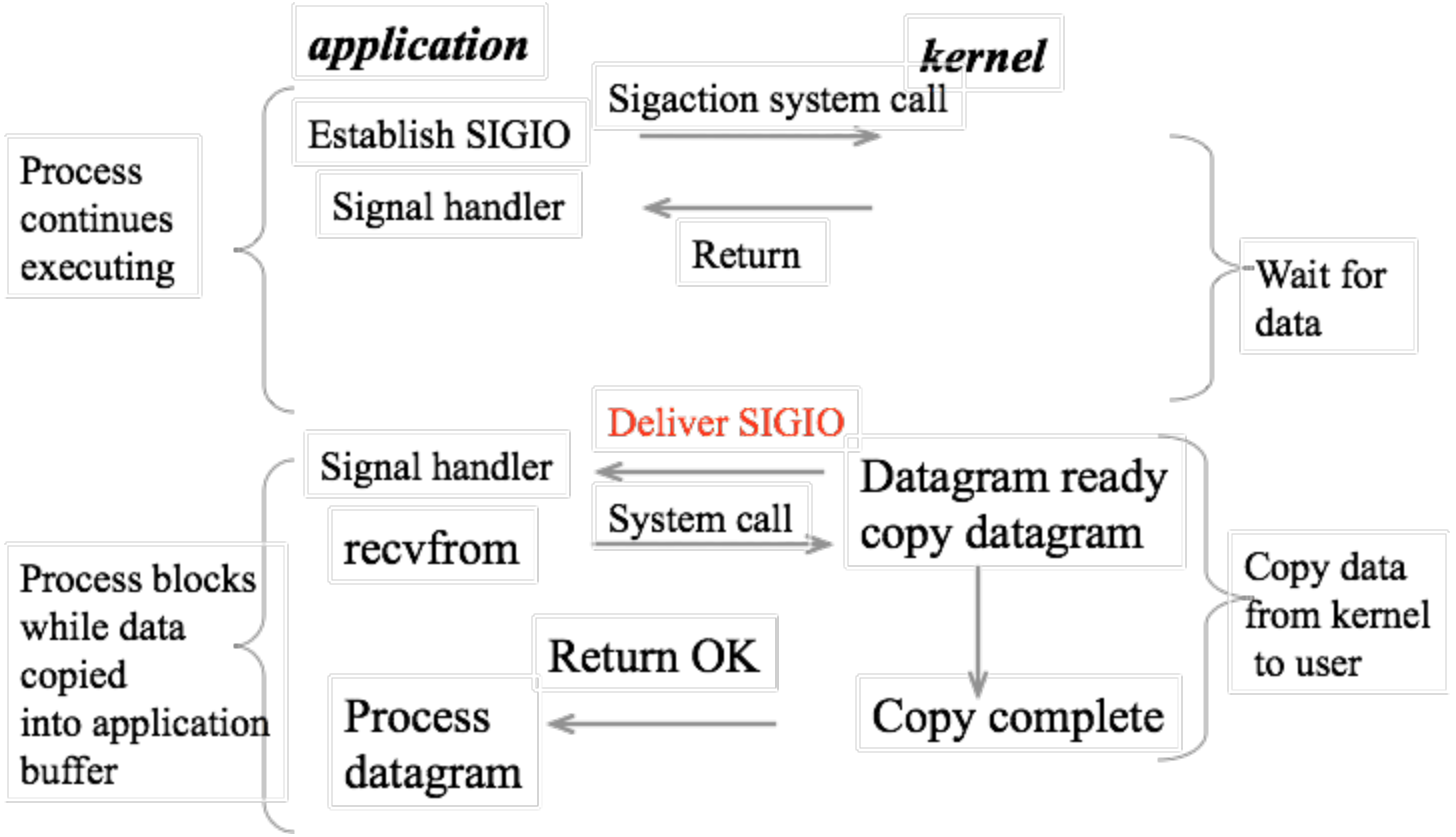
Vantagem em relação a E/S bloqueante?

Multiplexação de E/S

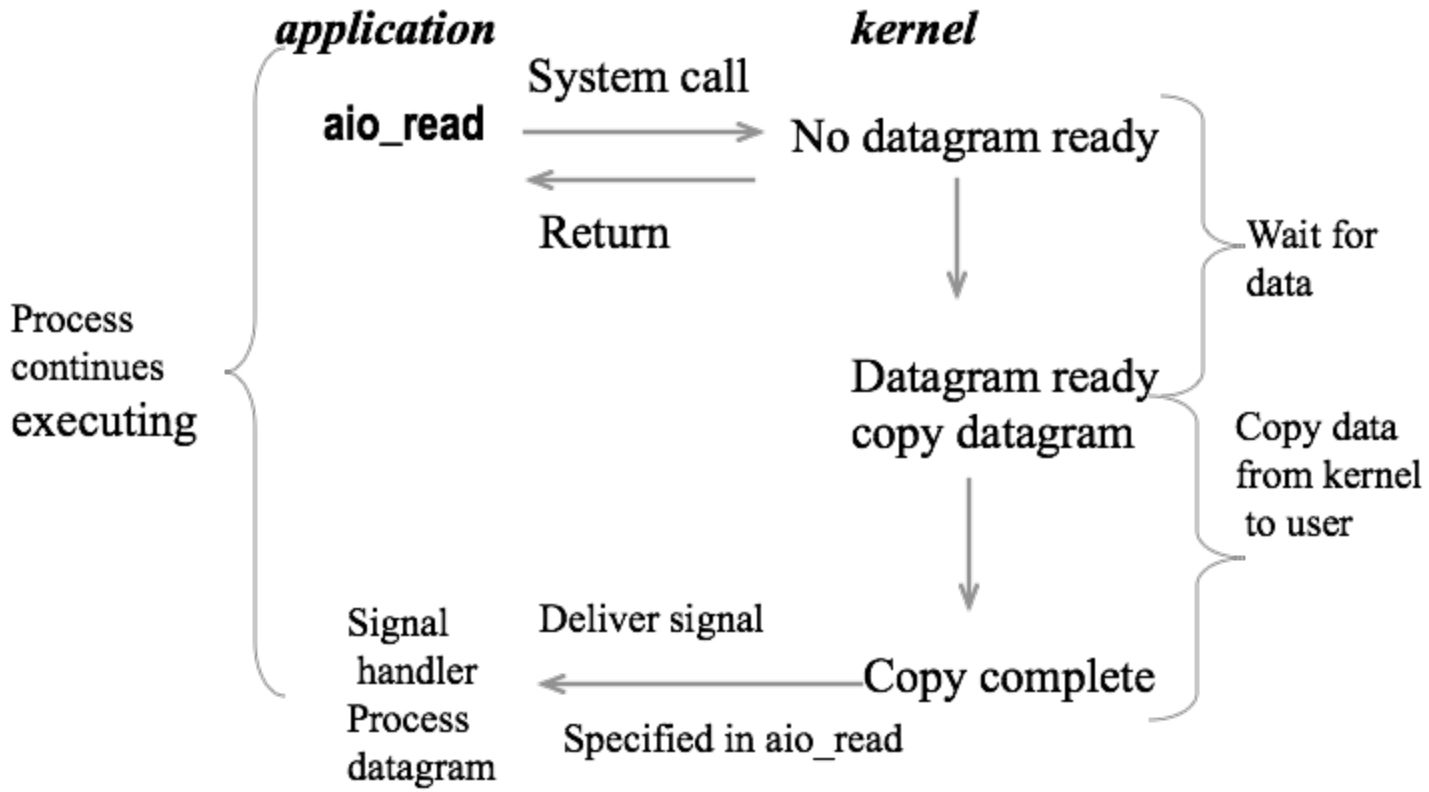


Vantagem: `select` pode ser usado para monitorar mais de um descritor!

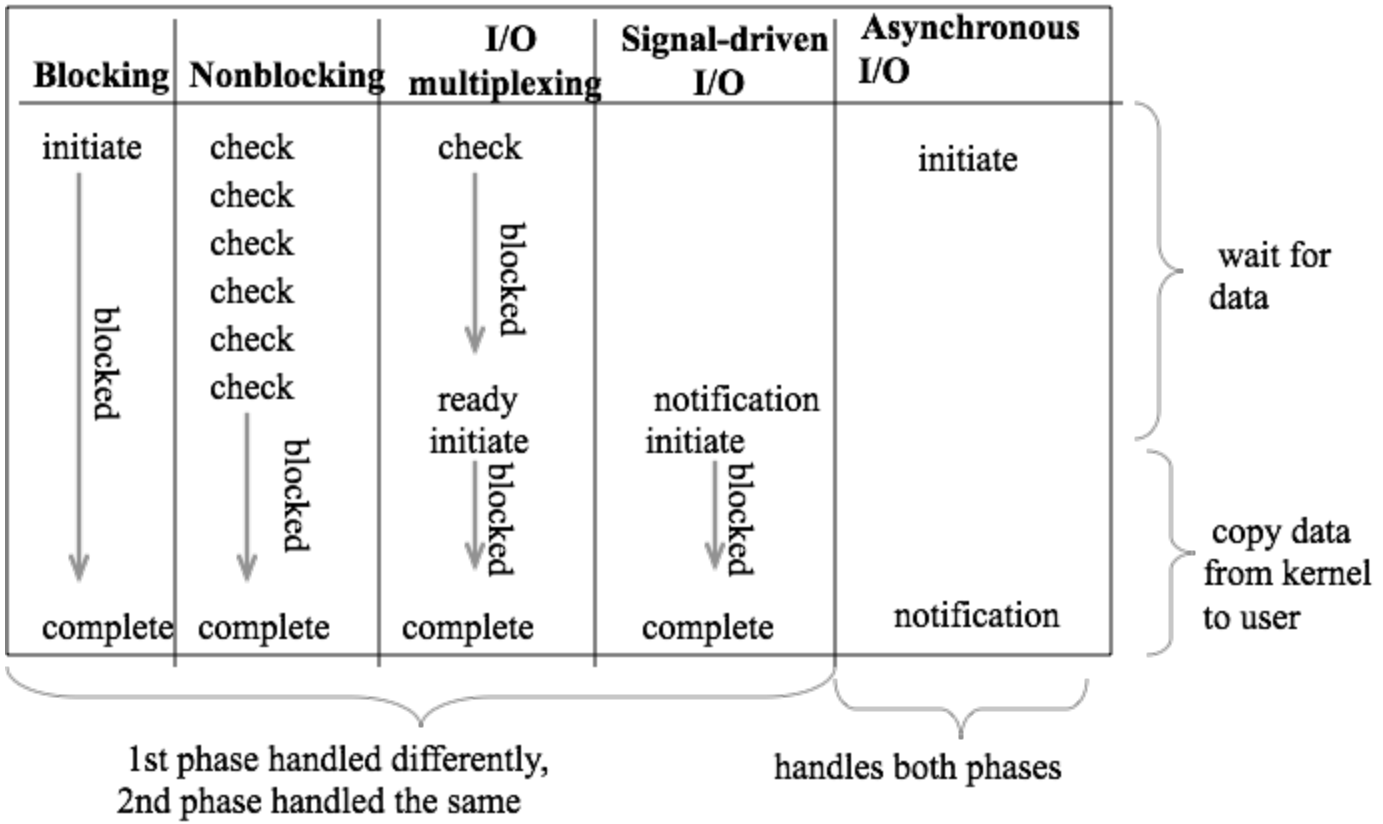
E/S orientada a sinal (SIGIO)



E/S Assíncrona



Comparação entre os modelos de E/S



Multiplexação de E/S - função `select`

- Instrui o kernel para “acordar” o processo quando um entre um conjunto de eventos ocorrer ou quando um certo intervalo de tempo tiver ocorrido.

- Especifica qual descritor se tem interesse (leitura, escrita ou exceção)

Multiplexação de E/S - função `select`

```
#include <sys/select.h>
#include <sys/time.h>

int select (int maxfdp1, fd_set *readset, fd_set
*writeset,  fd_set  *exceptset,  const  struct
timeval *timeout);
//Retorna: num. de descritores prontos, 0 se
timeout, -1 se erro

struct timeval{
  long  tv_sec;  /* segundos */
  long  tv_usec; /* microsegundos */ }
```

Multiplexação de E/S - função `select`

Três possibilidades de espera:

- Espera indefinidamente: retorna apenas quando um dos descritores especificados está pronto para E/S (`timeout = NULL`).
- Espera por um tempo fixo: retorna quando um dos descritores especificados está pronto para E/S, mas não espera por um intervalo maior do que definido pelo argumento `timeout`.
- Não espera: retorna imediatamente após verificar os descritores - *polling* (`tv_sec = 0, tv_usec = 0`)

Multiplexação de E/S - função `select`

```
int select (int maxfdp1, fd_set *readset, fd_set
*writeset,  fd_set  *exceptset,  const  struct
timeval *timeout);
```

`fd_set`: conjunto de descritores que queremos monitorar - normalmente um vetor de inteiros, onde cada bit de cada inteiro corresponde a um descritor.

Multiplexação de E/S - função `select`

Macros para manipular os conjuntos de descritores:

```
void  FD_ZERO(fd_set *fdset); /* "desliga" todos
os bits in fdset */

void  FD_SET(int fd, fd_set *fdset); /*"liga" o
bit para fd no fdset */

void  FD_CLR(int fd, fd_set *fdset); /*"desliga"
o bit para fd no fdset */

int   FD_ISSET(int fd, fd_set *fdset);/* o bit
para fd está "ligado" no fdset ? */
```

Multiplexação de E/S - função `select`

Macros para manipular os conjuntos de descritores:

```
void  FD_ZERO(fd_set *fdset); /* "desliga" todos
os bits in fdset */

void  FD_SET(int fd, fd_set *fdset); /*"liga" o
bit para fd no fdset */

void  FD_CLR(int fd, fd_set *fdset); /*"desliga"
o bit para fd no fdset */

int   FD_ISSET(int fd, fd_set *fdset);/* o bit
para fd está "ligado" no fdset ? */
```

Não esqueça de inicializar os conjuntos! (FD_ZERO)

Multiplexação de E/S - função `select`

```
int select (int maxfdp1, fd_set *readset, fd_set
*writeset,  fd_set  *exceptset,  const  struct
timeval *timeout);
```

`maxfdp1`: descritor máximo que deve ser testado + 1

Ex.: se foram "ligados" os bits para os descritores 1, 4 e 5, `maxfdp1 = 6`