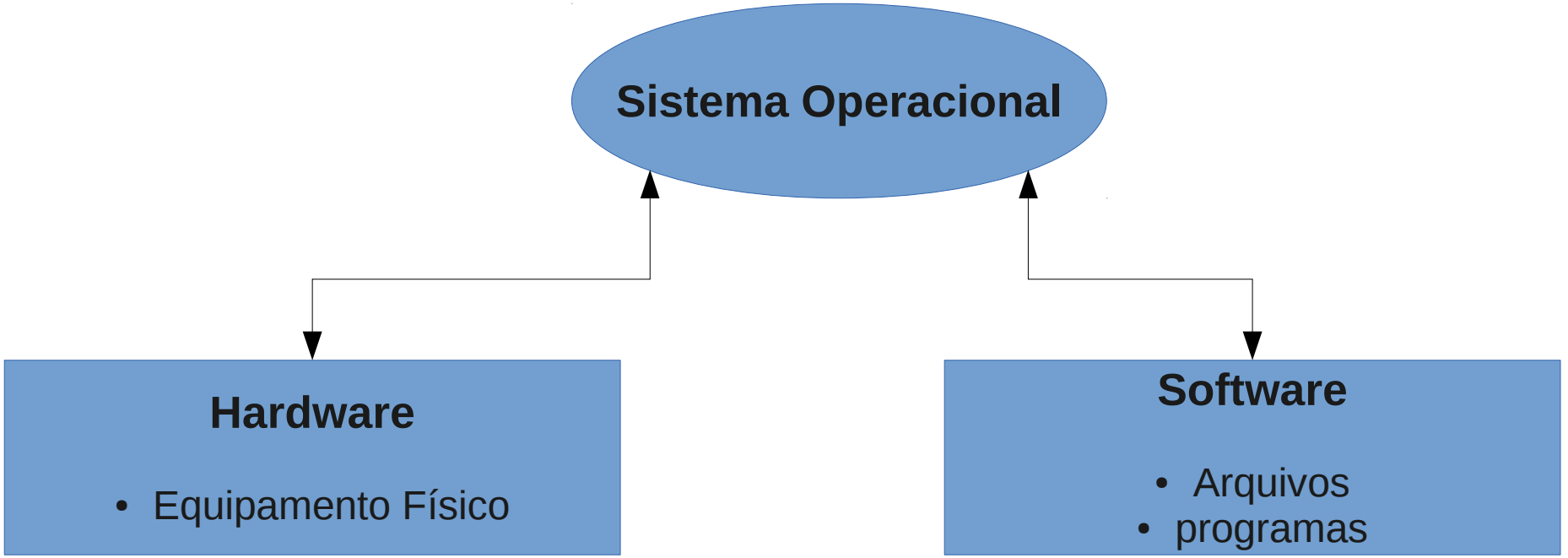
Software / Hardware



Sistema Operacional (SO)

- Faz ligação entre programas, hardware e usuário(s)
- Reconhece dados de entrada (teclado/mouse)
- Controla dispositivos de saída (tela)
- Gerencia sistema de arquivos
- Controla periféricos (impressora, etc)
- Cuida da alocação de memória e fila de execução de programas

Tipos de SO

- (a) Multi-usuário
- (b) Multi-tarefa
- (c) Multi-processos
- (d) Multithreading
- (e) Tempo Real

Exemplos:

- DOS, Windows
- OS/2
- **Linux**

Linux

- Baseado no Unix (Ken Thompson – Bell Labs – 1969)
- Criado por Linus Torvalds (kernel) - 1991
- Free Software Foundation
  - Software Livre (Licença GPL)
- Trabalho colaborativo
- Diferentes distribuições disponíveis

Arquitetura do Linux

- Kernel:
  - Suporte a drivers de diversos dispositivos
  - Gerenciamento de processamento e memória
- Shell e GUI (interface ao usuário)
- Utilitários de sistema:
  - Funções básicas para arquivos, como cópia, remoção, criação, etc.
- Programas

Sistema de diretórios

- / - diretório raiz
- /home - arquivos de usuários
- /bin - binários
- /etc - arquivos de configuração
- /var - logs e arquivos usados pelo sistema
- /lib - bibliotecas
- /dev - acesso a dispositivos físicos
- /home/<user> - diretório de trabalho do usuário <user>

Introdução ao Terminal

- Herança do Unix
- Permite gerenciamento básico do sistema e de arquivos e programas
- É necessário a realização de login do usuário

- Login:

Regras da linha de comandos

- Comandos são “case-sensitive”
- O retorno da linha de comando só ocorre depois que o comando for terminado – com sucesso ou não (*obs)
- O efeito de diferentes comandos pode ser afetado pelo uso de “flags”
  - Ex: “*ls -l*”
- Informações adicionais podem ser obtidas por:
  - *man <comando>*
  - *<comando> -h*
  - *<comando> --help*

O interpretador shell

- Atua como intermediário entre o kernel e o usuário
- É um intepretador de linhas de comando
- Funções:
  - Completa comandos [tab]
  - Guarda histórico de comandos [seta para cima]
  - Suporte a variáveis (de usuário e de sistema)
  - Suporte a *scripts*
  - Estruturas de comando básicas (loop, if, etc...)

Principais Comandos

- *pwd*: retorna diretório atual
- *mkdir <arg>* - *rmdir <arg>*: cria - remove pasta <arg>
- *cd <arg>*: entra no diretório <arg>
- *ls*: retorna lista de arquivos e pastas no diretório atual
- *cp <arg1> <arg2>*: copia <arg1> para <arg2>
- *mv <arg1> <arg2>*: move <arg1> para <arg2>
- *rm <arg>*: remove <arg>

Se aparecerem avisos ou erros relacionados a arquivos, use o *pwd* para se certificar em qual pasta (diretório) você se encontra !!

Comandos úteis

- *echo <mensagem>*: imprime <mensagem> na tela
- *who*: mostra quem está logado no sistema
- *tar cvf <nome> [arqs]*:
  - Concatena a lista de arquivos [arqs] no arquivo <nome>
- *tar xvf <nome>*:
  - Abre o arquivo <nome> nos seus arquivos constituintes
- *tar tvf <nome>*:
  - Lista os arquivos que constituem o arquivo <nome>
- *file <nome>:* retorna o tipo de arquivo de <nome>

O comando tar preserva a estrutura de diretórios !!

Coringas

- “*”: substitui nenhum ou qualquer caracter
- “?”: substitui apenas um caracter
- “[a-b]”: substitui apenas um caracter, dentro dos limites “a” e “b” onde, “a” e “b” podem ser:
  - Números → ex: [0-9]
  - Letras → ex: [a-z]

Mostrando o conteúdo de arquivos

- *clear*: limpa a tela
- *cat* <*arq*> : mostra o conteúdo TOTAL de <arq>
- *more* <*arq*> : mostra o conteúdo de <arq>, uma página por vez [espaço – b]
  - Pressione “q” para sair
- *head* <*arq*> : mostra primeiras 10 linhas de <arq>
- *tail* <*arq*> : mostra as últimas 10 linhas de <arq>

Busca dentro de arquivos

- *less <arq>*: similar a more
  - */'xxx'* : encontra a 1a ocorrência de 'xxx'
  - '**b**' e '**n**': navegação nas ocorrências de 'xxx'
- *grep 'xxx' <arq>*: lista as linhas que contém 'xxx' em <arq>
- *wc <arq>*: conta o tamanho de <arq>, o número de linhas e palavras no mesmo
- *wc -l <arq>*: conta o número de linhas no <arq>

Redirecionamento de saída

Ex:

- *cat* <*ENTER*>: editor in-line
- *cat* > *arq*: apaga conteúdo de <arq> e copia novo conteúdo para o mesmo
- *cat* >> *arq*: adiciona conteúdo em <arq>

Muito cuidado ao redirecionar as saídas !!

Este procedimento pode apagar de forma indesejada os dados de um arquivo !

Redirecionamento de Entrada

Ex:

- *sort*: organiza lista de elementos
- *sort* < *arq.txt*: organiza elementos encontrados no arquivo arq
- *sort* < *arq.txt* > *saida.txt*: organiza elementos encontrados no arquivo arq e escreve saída para arquivo saida.txt

Pipes

Concatena saída de comandos

Ex:

- *who*: mostra quem está logado no computador
- *who* | *sort*: organiza lista de usuários logados
- *who* | *wc -l*: mostra número de usuários logados

Para que um pipe funcione é necessário a execução de algum comando anterior ao pipe.
Os comandos devem estar na mesma linha !!

Atributos de arquivos

```
$ ls -l
total 18580
drwxr-xr-x 48 agenor agenor    4096 Fev 24 17:29 .
drwxr-xr-x  3 root   root      4096 Ago  8  2013 ..
-rw-r--r--  1 agenor agenor 1502889 Fev 18 13:17 .0gbas.so_u
```

-

Atributos:

D: diretório

R: read (leitura)	
W: write (escrita)	Usuário
X: execution (execução)	
R: read (leitura)	
W: write (escrita)	Grupo
X: execution (execução)	
R: read (leitura)	
W: write (escrita)	Outros
X: execution (execução)	

Mudando atributos de arquivos

- *chmod [ugoa][+-][rwx] <arq>*:

  onde:

símbolo	significado
u	usuário
g	grupo
o	others
a	all
+	adiciona flag
-	remove flag
r	leitura
w	escrita
x	execução

Mudando atributos de arquivos

- *chmod [0-6][0-6][0-6] <arq>*:
  - Read: 4
  - Write: 2
  - Execution: 1
    - Exemplos:
      - Chmod 666 arq1 => -rw-rw-rw- arq1
      - Chmod 421 arq2 => -r---w---x arq2

Processos

- Qualquer comando ou programa rodado no linux têm um número de identificação (PID)
- *ps*: lista processos ativos pelo usuário
- *ps aux*: lista TODOS processos ativos

```
agenor@pcl216 ~ $ ps
  PID TTY          TIME CMD
 2303 pts/0    00:00:00 bash
 2513 pts/0    00:02:51 firefox
 2644 pts/0    00:00:06 plugin-containe
 3098 pts/0    00:00:00 ps
```

-

Processos em *background*

- <*comando*> *&*: roda programa em background
- <*comando*> → *^z [CTRL+z]* → *bg*:
  - Roda programa em background, depois de te-lo iniciado em foreground
- *jobs*: lista de programas chamados diretamente pelo usuário
- *fg %[número]*: reinicia programa na linha [numero] da lista de trabalhos, que foi suspenso previamente pelo usuário.

Faça um bom uso da execução em background !
Ela se torna extremamente útil nas atividades de programação (e outras em que se use o terminal e um editor de textos).

Forçando o término de processos

- *kill %[numero]*: termina processo encontrado na linha [numero] da lista dada por *jobs*
- Alternativa:

```
agenor@pcl216 ~ $ ps
  PID TTY          TIME CMD
 2303 pts/0    00:00:00 bash
 2513 pts/0    00:02:51 firefox
 2644 pts/0    00:00:06 plugin-containe
 3098 pts/0    00:00:00 ps
agenor@pcl216 ~ $ kill -9 2513
```

-
-