# Système d'exploitation

# Unix

http://labsticc.univ-brest.fr/~lemarch/FR/Cours/

Laurent Lemarchand
LISyC/UBO
Laurent.Lemarchand@univ-brest.fr

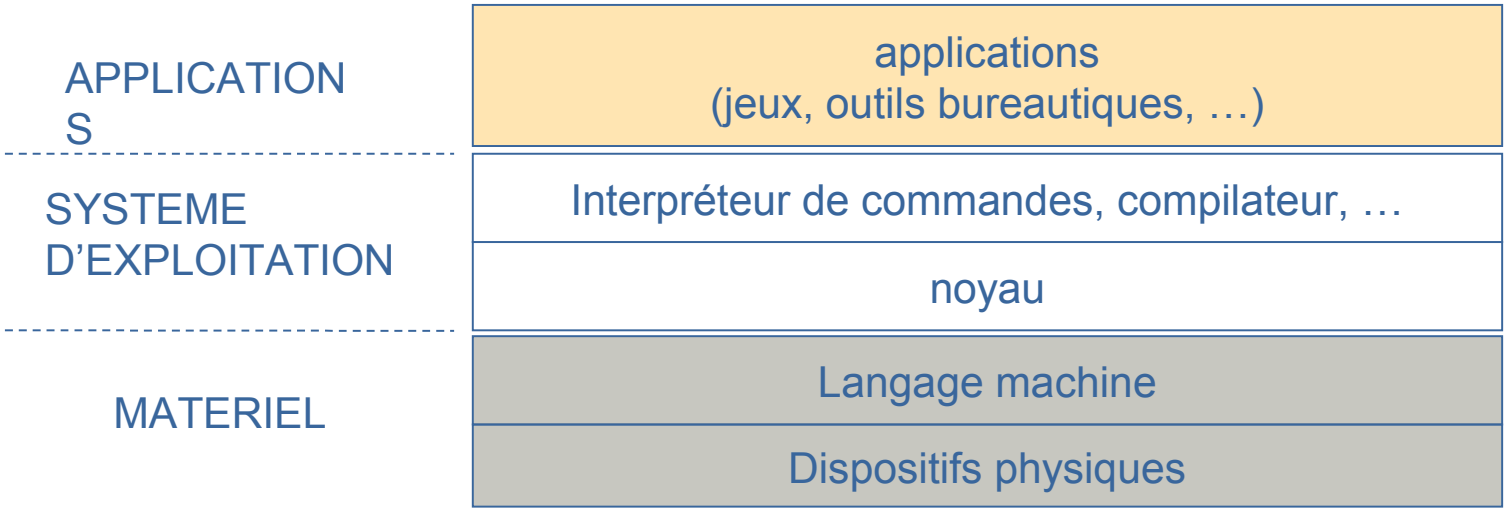
# Les systèmes d'exploitation

- C'est l'interface entre l'utilisateur et le matériel
- Ses fonctions principales sont :
  - Contrôle des ressources (allocation et gestion du CPU et de la mémoire)
  - Contrôle des processus
  - Contrôle des périphériques
  - …
- Il contient des outils de gestion utilisables par les applications, tels que la manipulation de fichiers, gestion d'impressions, date…

# Les systèmes d'exploitation

- Exemples:
  - Unix, DOS, Windows, Mac OS, Linux, OS/2, BSD, ...

- Architecture-type:



# Unix/Linux

- Propriétés
  - multi-tâches
  - multi-utilisateurs
  - multi-postes
  - Parfois libre (et gratuit)  !!

# Unix/Linux - Historique

1969: Ken Thompson (Bell labs / MIT) : volonté d'avoir un système multitâches et multiutilisateurs, fiable pour être utilisé par des laboratoires de recherche.

1973: écriture en C -> système portable

1974-1977: distribution aux universités

1979: AT&T commercialise UNIX

Apparitions des UNIX propriétaires

ULTRIX (DEC), AIX (IBM), HP-UX

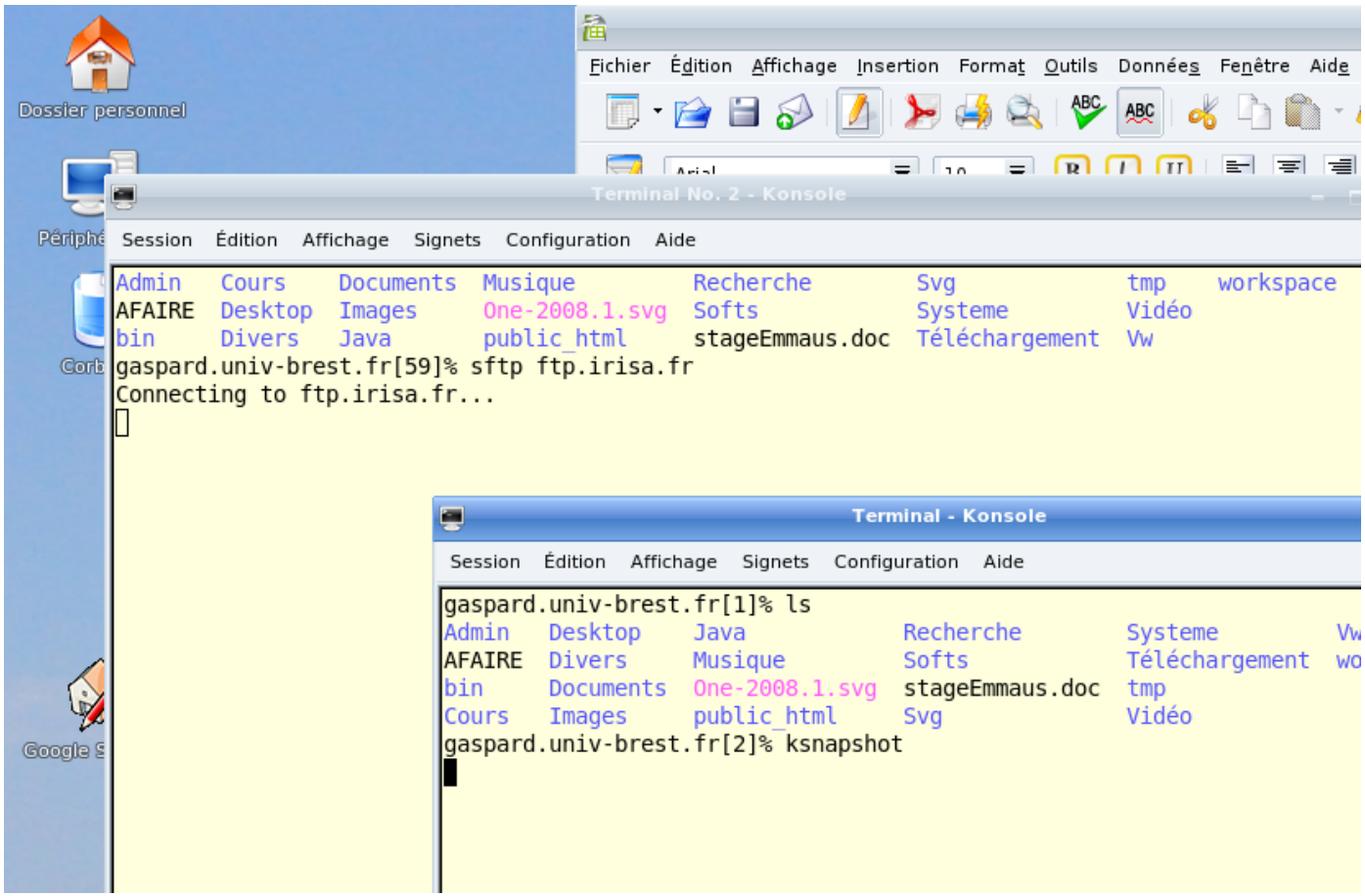
1987 : A.S. Tanenbaum : Minix en cours d'informatique

1990 : Larry Wall : Perl pour la gestion système

1991 : Linus Torvalds (étudiant finlandais de 21 ans) : LINUX basé sur Minix

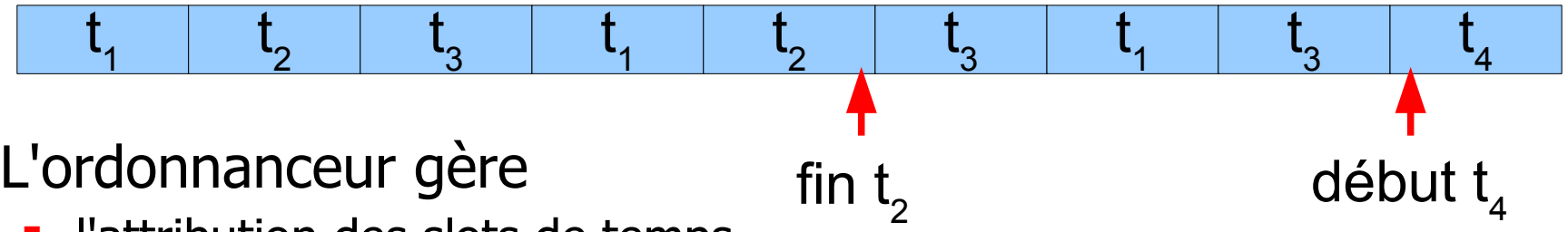
# Unix/Linux : propriétés

- Multi-taches
  - <> DOS
  - Plusieurs programmes ou commandes exécutées simultanément



# Multi-tâche

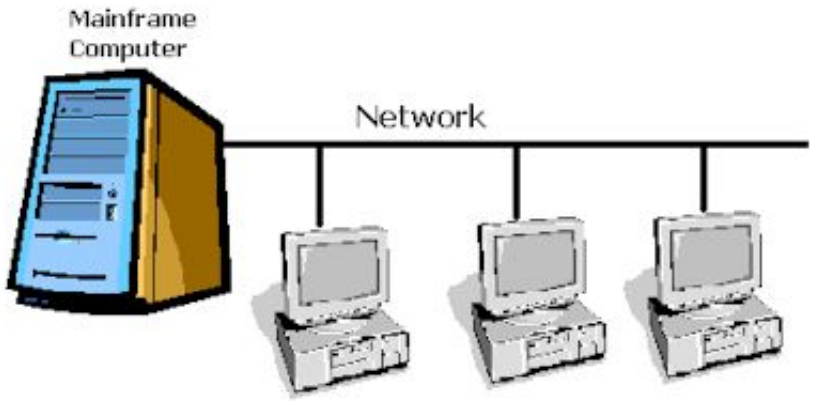
- L'ensemble des activités (tâches) se déroule simultanément
  - Chaque tâche a droit à un temps de calcul
  - Le temps est découpé en tranches
  - En apparence, les tâches s'exécutent en parallèle
  - Tire partie de la lenteur des tâches / possibilités de l'ordinateur




- L'ordonnanceur gère
  - l'attribution des slots de temps
  - les changement de contexte et de mémoire
  - Les priorités des tâches

# Multi-utilisateurs

- Les tâches peuvent appartenir à des utilisateurs différents




- Connexions à distance
- Taches système
- Séparation des espaces de travail
  (notion de propriétaire pour les données et les pages mémoire)

# Invites de commandes (1)

- Il existe différents types de shell (invite) :
  - Bourne shell sh ~ 1975
  - C shell csh
  - Korn shell ksh
  - Bourne again shell bash
- Lancé dans un terminal
- Interaction en mode commande avec le système
  ATTENTION SENSIBLE A LA CASSE

# Invites de commandes (2)

- Le shell présente un prompt et attend des commandes
- Il reprend la main après l'exécution du processus associé à l'exécution de la commande demandée
- Les commandes ont un nom, des arguments et des options

```
ls
ls -l
ls -l MonDirectory
ls Mondirectory -la
```

# Connexion sur une machine

- Ouverture de la session de travail
  - Authentification de l'utilisateur:
    - Login (username)
    - Password (mot de passe)

  - Nouvelle connexion :

    **`su lemarch`**

    par exemple pour changer d'utilisateur dans un terminal

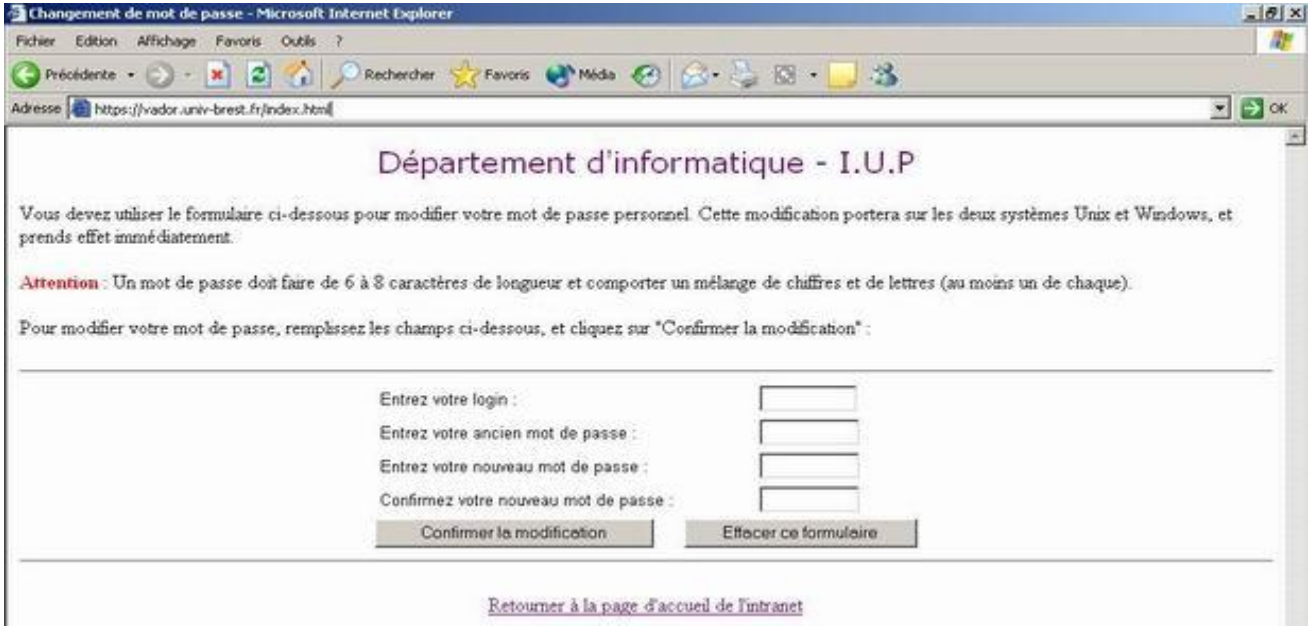
# Les mots de passe

- Les mauvais mots de passe (dictionnaire)
  - titi
  - albatros
  - rex
- Un bon mot de passe
  - ja!m34r)%
- Changer régulièrement
  commande **`passwd`**

# Les mots de passe au département

- Connexion sur https://vador.univ-brest.fr/index.html

- Mot de passe Unix et Windows



# Lancement d'un terminal

- Pour utiliser une machine, il faut se mettre en relation via un TERMINAL avec le système
  - Terminal physique (ex. vt100)
  - Terminal virtuel (ssh, telnet)

- Tâches au démarrage du terminal :
  - type de terminal
  - lance un interpréteur de commande (shell)
  - définit le clavier comme entrée standard
  - définit l'écran comme sortie standard
  - fichiers .login, .cshrc pour définir des variables d'environnement : PATH, GROUP, TERM ...

# Variables d'environnement

- Exemples
  - PATH : chemins vers les exécutables
  - TERM : type de terminal (clavier)
  - DISPLAY : écran d'affichage
- Gestion
  - **`env`**
  - **`setenv PATH $PATH:"monrep"`**
  - **`echo $USER`**

# Fichier .cshrc au département

- http://intranet-depiup.univ-brest.fr/faq/csh_login.htm

```
# .cshrc fichier execute' lorsqu'un shell csh ou tcsh est lance' et avant le .login
setenv MANPATH /usr/man:/usr/local/man:/usr/share/man:/usr/dt/man
setenv PATH "/usr/dt/bin/:/usr/local/bin:/usr/openwin/bin:/opt/bin:/opt/sfw/bin"
setenv PATH "/usr/local/sbin:/usr/sbin:/sbin:${PATH}:${HOME}/bin:/opt/prolog/kit45:."
setenv PATH "/usr/local/j2sdk1.4.1/jre/bin":${PATH}
# Environnement Eclipse
setenv CLASSPATH
setenv CLASSPATH ${CLASSPATH}:"/usr/local/eclipse/startup.jar"
# Définition des paramètres d'environnement
set host=`hostname`
set prompt= ( `echo $host`"[\!]% " )
set history = 25
set ignoreeof noclobber notify nonomatch listpathnum
set filec
set correct=all
```

```
# Définition des alias

alias rm 'rm -i'
alias cp 'cp -i'
alias mv 'mv -i'
alias h history
alias ll /bin/ls -lg
```

# Fichier .login au département

- http://intranet-depiup.univ-brest.fr/faq/csh_login.htm

```
# ------------------------------------------------------------
# Si vous voulez ajouter des instructions particulieres a ce script il vous est vivement conseille de ne PAS MODIFIER le
    fichier
# directement mais plutot de travailler sur le script .monlogin qui est à considérer comme votre script personnel, et qui
    est
# lancé à la fin de celui-ci, et dans lequel vous pouvez faire ce que vous voulez...
# Ainsi en cas de probleme, il vous suffira de commenter la dernière ligne pour revenir a l'etat initial.
#
# Systeme d'exploitation
set TYPE_SYSTEME = ( `uname -rs` )
# Texte de bienvenue
echo " "
echo "Bonjour $USER, bienvenue au departement d'informatique - I.U.P"
date '+Nous sommes le %A %d %B %Y et il est %kH%M.'
echo "\
Votre systeme est :
    $TYPE_SYSTEME\
Votre terminal est un : $term \
Votre home-directory est : $cwd\
"
cat /home2/applis/motd

source .monlogin
```

# Les fichiers (1/3)

- Fichier : enregistrement sur le disque contenant des informations

- 2 types:
  - texte (le codage du texte change d'une machine à une autre : transfert de fichiers )
  - Autres = code binaire
    - Images (tiff, jpg, gif...)
    - Code objet
    - ...

# Les fichiers (2/3)

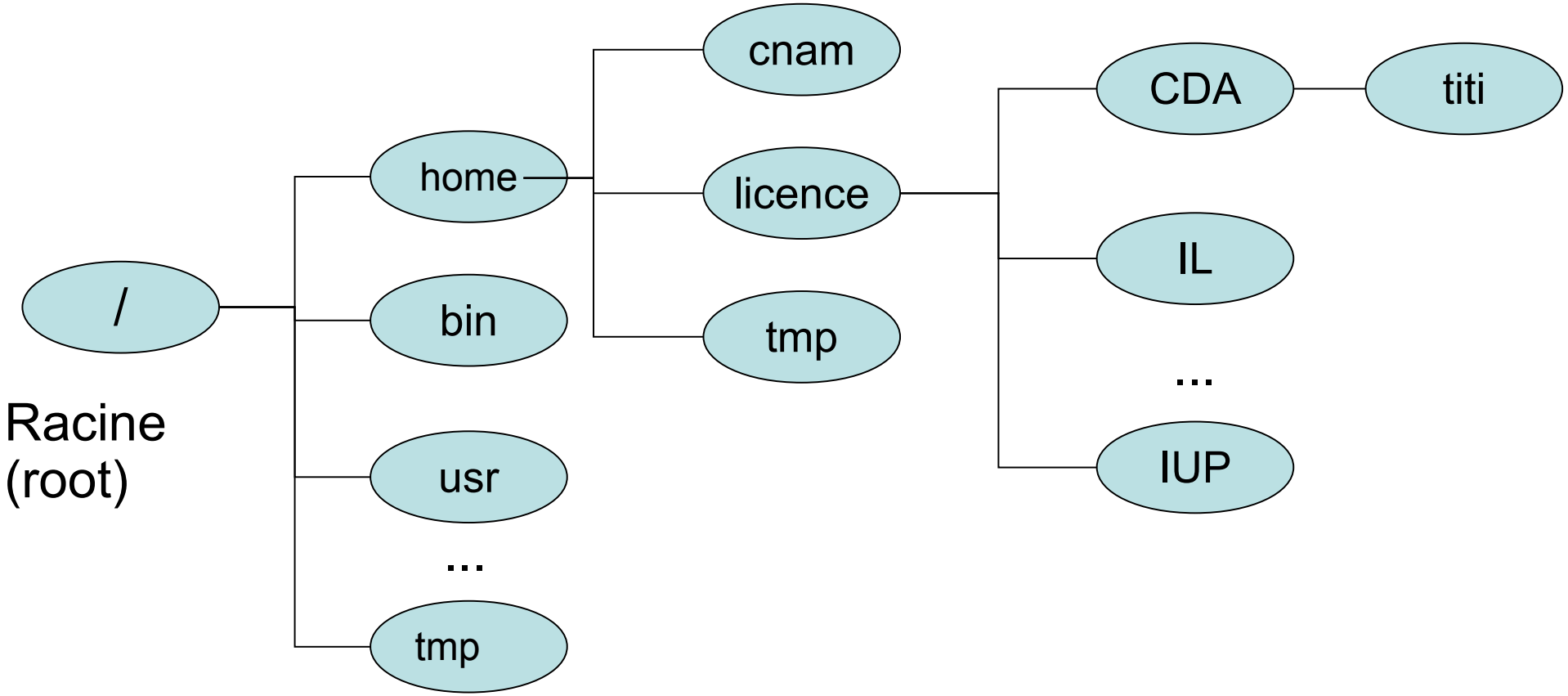
- Un fichier est identifié par son nom
  - Caractères autorisés: a-zA-Z0-9 - _
  - Caractères spéciaux :
    - Espace . , / : * & ; % ` " \ $ ()[]{}+=<>

- exemples:
  - budget92, budget92.doc
  - Budget_2003.exc
  - notes_biophy_2003.txt

# Les fichiers (3/3)

- l'extension (caractères après le point) informe sur l'application qui a créé le fichier
  .c .o .ps .tar .gz .txt ....
- date de création du fichier
- taille en octets
- droits d'utilisation

# L'arborescence du système de fichiers UNIX



# L'arborescence du système de fichiers UNIX

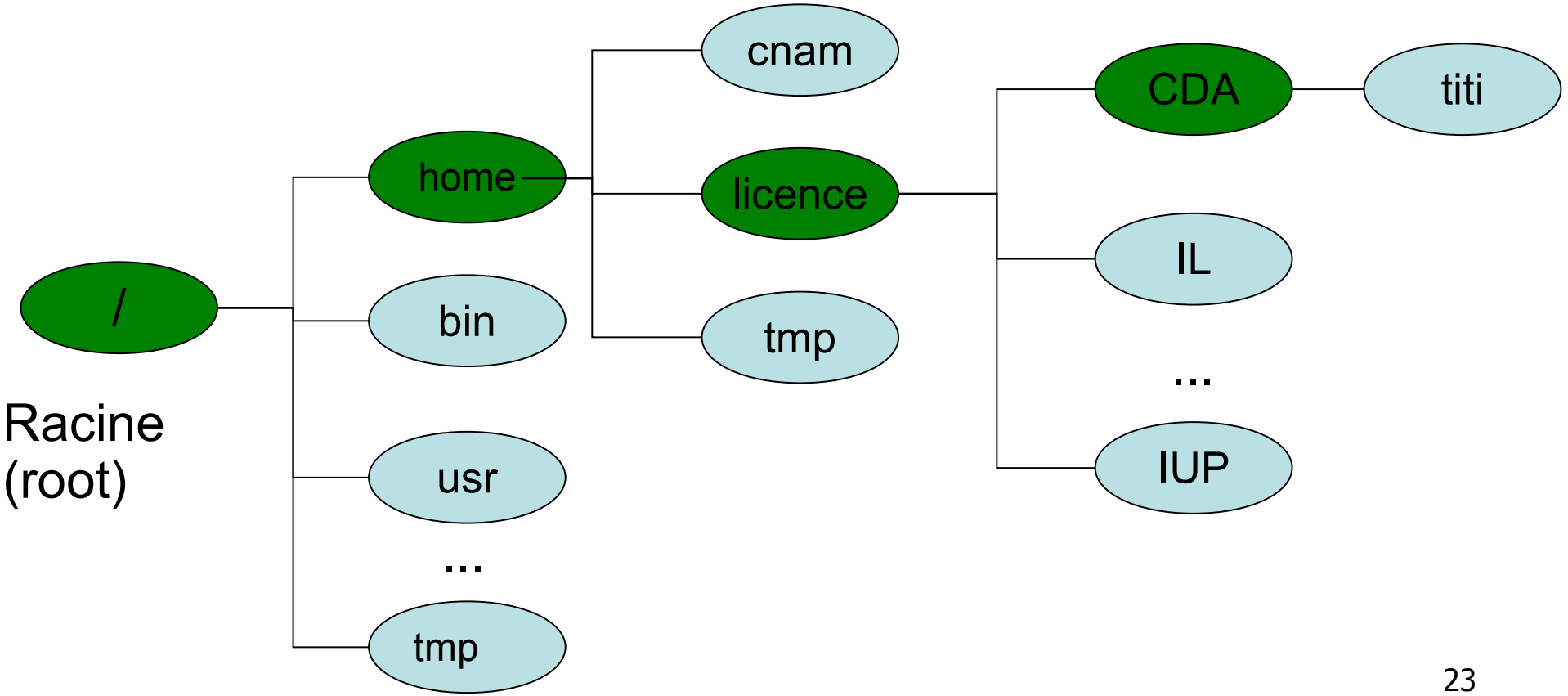
- Un fichier : flot d'octets (vue uniforme sur tous les périphériques d'I/O)

- Structure hiérarchique
  - Racine (root) notée / (slash)
  - Arborescence de répertoires
  - Répertoires utilisateurs (homedir ~)

- Répertoire courant d'exécution des processus
  ex: pour un shell, commande **pwd**

# Chemin d'accès (1/2) absolu

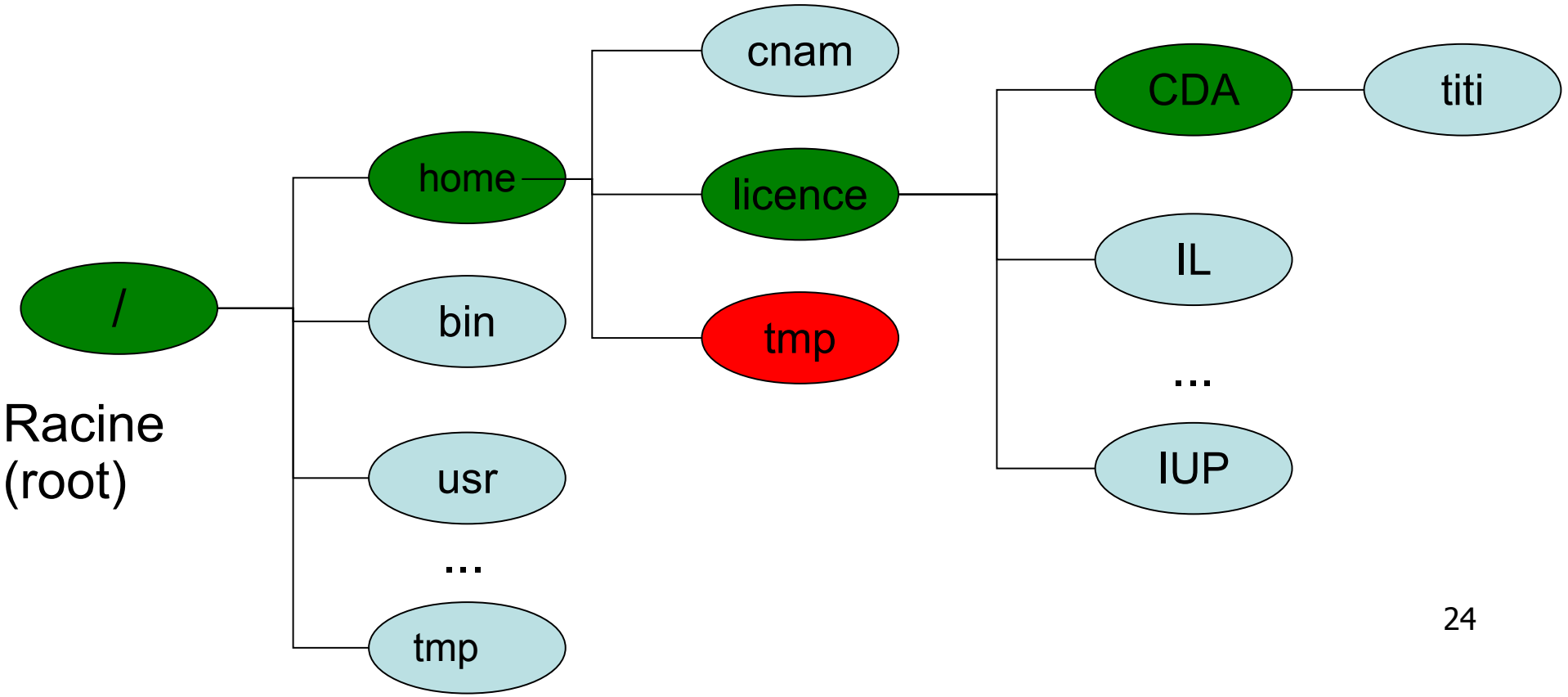
- À partir de la racine
  **`/home/licence/CDA`**
  **`//tmp/bidule\ truc/machin`**



# Chemin d'accès (2/2) relatif

- À partir du répertoire courant

```
../licence/CDA
../tmp/.././../home/licence/CDA
~titi/..
```



# Commandes de gestion de fichiers

- **pwd** (Print Working Directory) donne le chemin du répertoire courant

- **ls** (LiSt) (options -l et -a) liste des fichiers et répertoires dans le répertoire courant ou le répertoire donné en argument

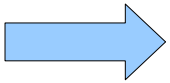
- **cd** (Change Directory) change de répertoire
  **cd Tests** pour aller dans le répertoire Tests
  **cd** pour aller dans le répertoire par défaut de l'utilisateur (**cd ~user**)
  **cd ..** pour remonter l'arborescence

# Commandes de gestion de fichiers

- **`touch`** crée un fichier vide
- **`rm`** efface des fichiers. **`rmdir`** efface des répertoires vides
- **`mv`** renomme fichiers et répertoires
- **`cp`** copie des fichiers remonter l'arborescence
- **`ln`** `crée un nouveau lien vers un fichier`

# Commandes de gestion de fichiers

- **`find`** pour la recherche dans l'arborescence
- **`diff`** pour comparer des fichiers
- **`cat`** pour les afficher sur la sortie standard
- **`df`** pour lister les partitions



**`man`** pour le manuel des commandes en ligne

# Génération de noms de fichiers avec le shell

- Avant toute exécution de commande, le shell effectue la substitution des caractères spéciaux suivant par les noms de fichiers qui y correspondent

- `*` suite quelconque (peut être vide)
- `?` un caractère
- `[...]` un des caractères entre crochets

```
% ls *
fic.c fic.o fiche a.out

% ls *o
fic.o

% ls fic.?
fic.c fic.o

% ls fic.[co]
fic.c fic.o

% ls /etc/ld.*
ld.so.cache ld.so.conf

% cat /u*/inc*/ti*.h
/usr/include/time.h
...
```

# Droits d'accès

- **`ls -l`** pour visualiser les droits
  - En lecture (r)
  - En écriture (w)
  - En exécution (x)

```
drwxr-xr--   1 lemarch  lib          0 Jun  6 14:25 Stmp
-rwxr--r--   1 lemarch  lib     859053 Feb  5  1999 util
-rw-r--r--   1 lemarch  lib      99040 Mar  2  1999 poly.tar.gz
```

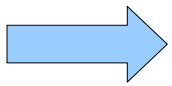
- **`chmod droits fichiers`** pour gérer les droits
  - Du propriétaire (u)
  - Du groupe (g)
  - Du reste du monde (o)

```
chmod go+x util
```

# Archivage

- **`tar`** pour gérer une archive
  **`tar <commandes> <archive> <fichiers>`**
  - Création : **`tar cvf archi.tar Rep f1 f2`**
  - Listing : **`tar tvf archi.tar`**
  - Extraction : **`tar xvf archi.tar`**

- **`gzip bzip compress`** pour compresser une archive
  **`gzip archi.tar`**



**`archi.tar.gz`**

# Gestion des processus (1/3)

- La commande **`ps -aux`** permet d'obtenir la liste des processus en cours sur une machine

- Un processus est identifié par un numéro: le PID (Process Identificator)

- Un processus appartient à un utilisateur (root si c'est un processus système)

# Gestion des processus (2/3)

- Un processus interactif utilise la sortie et l'entrée standard du terminal.

- On peut lancer une commande en tâche de fond grâce au caractère **&** :

```
prog &
```

- Le processus peut être  amené au premier plan avec la commande **fg** (en arrière plan avec **bg**)

# Arrêt des processus (3/3)

- Arrêt forcé d'un processus :
  **`kill -9 PID`**
  **`ctrl-C`** pour un processus interactif

- Interruption momentané d'un processus
  **`ctrl-Z`**, typiquement suivi de **`bg`**

# Flux de données

- On peut rediriger les flux de données (sortie standard / entrée standard) en utilisant des fichiers : redirections

- redirection d'entrée :
  **`commande < fichier`**

  jusqu'à un marqueur sur l'entrée standard :
  **`commande << marqueur`**

- Redirection de sortie :
  **`commande > fichier`**

  en ajout :
  **`commande >> fichier`**

# Enchainement de commandes

- On peut rediriger la sortie standard d'une commande en entrée d'une autre commande : enchainement de traitements, avec un ***pipe*** :

```
commande1 | commande2
```

- Exemples :

```
cat fic | wc -l
uuencode im.jpg | mail lemarch -s "trombi"
grep IUP listing.txt | grep licence | \
cut -d: -f1 | sort
```

# Quelques commandes

**`cal`** donne le calendrier d'une année

**`date`** donne la date

**`sort`** tri des données

**`head -n`** affiche les n premières lignes reçues

**`tail -n`** affiche les n dernières lignes reçues

**`grep`** affiche les lignes contenant une expression régulière indiquée en argument

**`cut`** sélectionne des colonnes dans une ligne

**`wc`** compte le nb de caractères, mots, lignes

**`time programme`** durée d'exécution de programme

**`which programme`** localisation d'un programme