UNIVERSITÉ DE LORRAINE | FACULTÉ DES SCIENCES ET TECHNOLOGIES

Licence 2 informatique

Outils système

SE, fichiers, processus & make

Alexis Scheuer

Maître de conférences UL

FST, dépt info. & Loria

Plan

- SE & gestionnaires de fenêtres

  un tour d'horizon
- Gestion des fichiers

  systèmes de fichiers, commandes & droits
- Gestion des processus

  états et commandes
- Génération de fichiers

  make, automake, autoconf, cmake, ...

Premiers ordinateurs et SE

- Années 40 : calculateurs → ordinateurs
- Années 50 : premiers systèmes d'exploitation
- 1965 : apparition du terme OS (→ DOS)
- 1969 : Unics (précurseur d'Unix)
- 1981 : MS-DOS
- 1985 : Windows 1.0
- 1991 : Linux 0.01

Premières interfaces graphiques

- Xerox Alto (1973)
  - Xerox Star (1981)
  - Apple Lisa (1983)

Pourquoi 1984 n’a pas été “1984”

- Apparition du MacIntosh
  - 128 ko de RAM (16 SE, 22 CG, + 90) + 64 de ROM
  - écran 9", 512×342, NB
  - disquette 400 ko

- Atari ST
  - 320x200,16 – 640x200,4
  - disquette double densité
- Amiga, Commodore 64, …

Windows ?

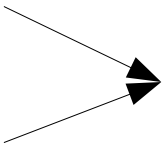
- Version 1.0 apparue fin 1985

  très limité (pas de superposition – brevet)

- Pas de succès avant 1990 & Windows 3.0

3.x → 95, 98, ME

NT (, 2000)



XP, Vista, 7, 8 & 10

- Devenu le système le plus répandu sur les PC

Windows : défauts !

- Manque de stabilité (amélioré)

  Erreur d'accès mémoire → écran bleu

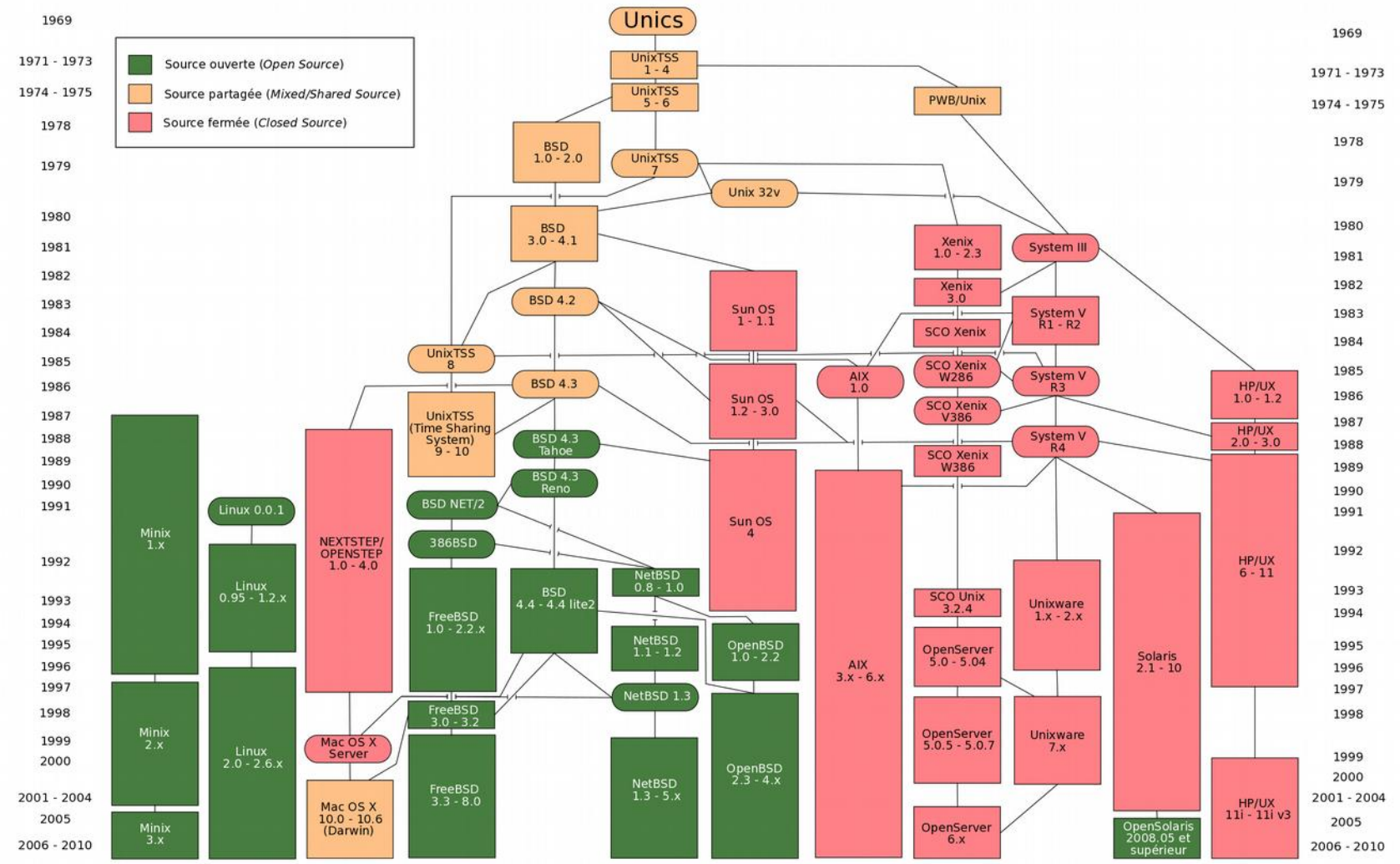
- Peu sécurisé

  Beaucoup de comptes administrateur + bugs

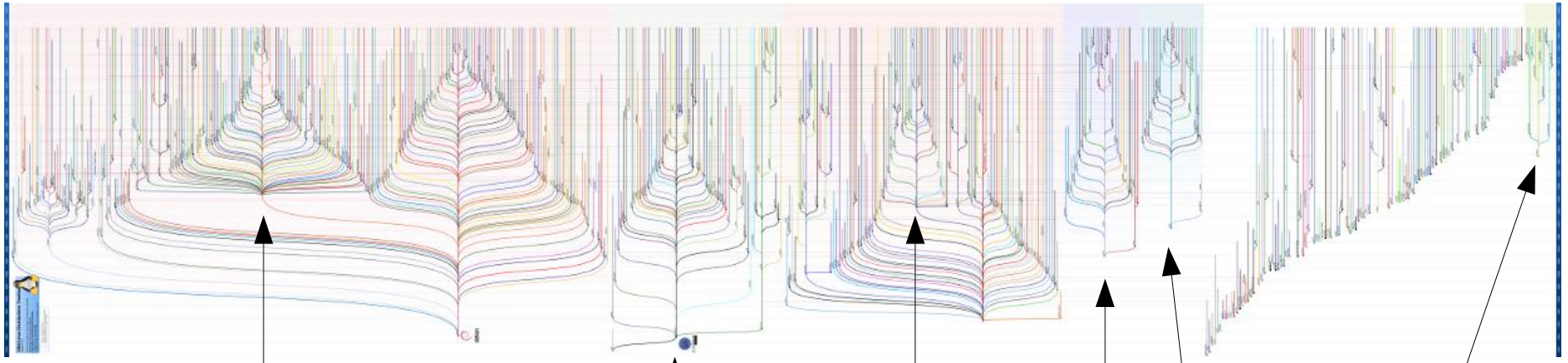
- Fermé, cher et espion

  Windows 10 officiellement un cheval de Troie

Unix : variantes & héritiers

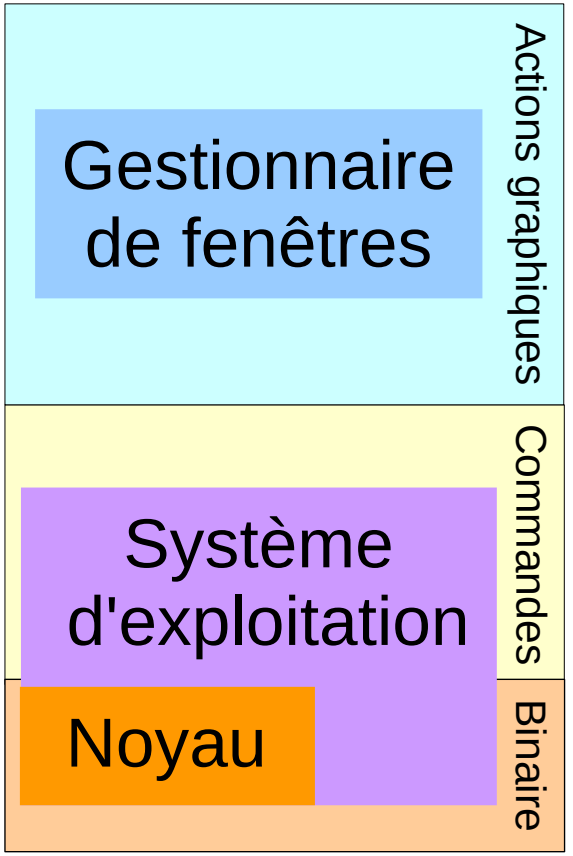


Linux : encore pire !



Gestionnaires de fenêtres

- Système de fenêtrage :
  X Window (1984)
- … → TWM (87)
  → FVWM (93) → XFWM (98),
  …
- Environnements de bureau :
  Gnome, KDE, Openbox,
  XFCE, Enlightment, …



Répartition des SE

- Ordinateurs personnels

  ~ 90 % Windows, 7 – 8 % MacOS, qq % Linux

- Téléphones

  < 1 % Windows, ~ 20 % IOS, ~ 75 % Android

- Serveurs

  30 – 40 % Windows, 30 – 35 % Unix, idem Linux

- Supercalculateurs

  100 % Linux

Plan

- SE & gestionnaires de fenêtres

  un tour d'horizon
- Gestion des fichiers

  systèmes de fichiers, commandes & droits
- Gestion des processus

  états et commandes
- Génération de fichiers

  make, automake, autoconf, cmake, ...

Gestion du disque

- Plusieurs systèmes sur un ordinateur ?
  - installations multiples & programme d'amorçage (*NT Loader*, *Lilo* ou *Grub*)
  - machine(s) virtuelle(s)
- Partitions
  - compartiments étanches du disque
  - indépendance entre ces parties (système)
  - outils : (g)parted

Systèmes de fichiers

- Très nombreux (peu sous Windows en natif)
  - Unix/Linux : ext, ext2, ext3, ext4, …
  - Windows : Fat (12, 16, 32, V, ex), NTFS, …
  - Périph. : DevFS, CDFS, ProcFS, UDev, SysFS, ...
  - Réseau : SMB, NFS, AFS, CIFS, SSHFS, …
- Nouveaux systèmes locaux journalisés
  - sauvegardes incrémentales
  - + rapide & + sûr

Représentation des fichiers

- Tous systèmes :
  ensemble de fichiers = graphe orienté (forêt)
- Fichier = nœud
  - feuille = fichier (document)
  - nœud non terminal = répertoire (fichier contenant une liste de descripteurs de fichier)
- Faible différence entre fichiers et répertoires

Nommage d'un chemin

- Séparateur
  - DOS & Windows : \
  - Unix, Linux & MacOS : /
- Chemin relatif / absolu
  - . = répertoire courant
  - .. = répertoire parent
  - racines multiples sur Windows & MacOS (lettre majuscule), unique sur Unix & Linux (/)

Fichiers : commandes (sh)

- Description

  ls = *LiSt*, pwd = *Print Working Directory*,
  cd = *Change Directory*, cat = *conCAT & display*

- Modification

  touch = change la date (ou crée un fichier),
  mkdir = *MaKe DIRectory*, cp = *CoPy*, mv = *MoVe*,
  rm = *ReMove*

  3 derniers : -f = *force*, -i = *interactive*, -n = *no clobber*, -r = *recursive* (sauf mv)

Fichiers : droits

- Unix/Linux : `chmod`
  - *r(ead)*, *w(rite)*, *(e)x(ecute)*
  - propriétaire (utilisateur), un groupe, autres
- NTFS : `icacls`, plus fin
  - DOS + *m(odify)*, *d(elete)*, *c(ontrol)*, *s(ecurity)*, …
  - autant d'utilisateurs ou de groupes que souhaité

Droits : affichage (sh)

Commande `ls` avec option `-l` (+ `-h`)

```
ajs@cornu:OS$ ls -alh
total 28K
drwxr-xr-x  5 ajs  larsen 4,0K janv. 27 11:29 ./
drwxrwxrwt 20 root root    12K janv. 27 11:30 ../
drwxrwxrwx  2 ajs  larsen 4,0K janv. 27 11:29 Commun/
-rw-r--r--  1 ajs  larsen    0 janv. 27 11:28 fich
-rw-rw----  1 ajs  larsen    0 janv. 27 11:29 groupe
drwx------  2 ajs  larsen 4,0K janv. 27 11:28 Perso/
drwxr-xr-x  2 ajs  larsen 4,0K janv. 27 11:28 Rep/
```

Droits : modification (sh)

Commande : `chmod` droits fichier

- droits
  - dans l'ordre, pour l'utilisateur, le groupe et les autres
  - représentation binaire : `r` = 4, `w` = 2, `x` = 1
- fichier

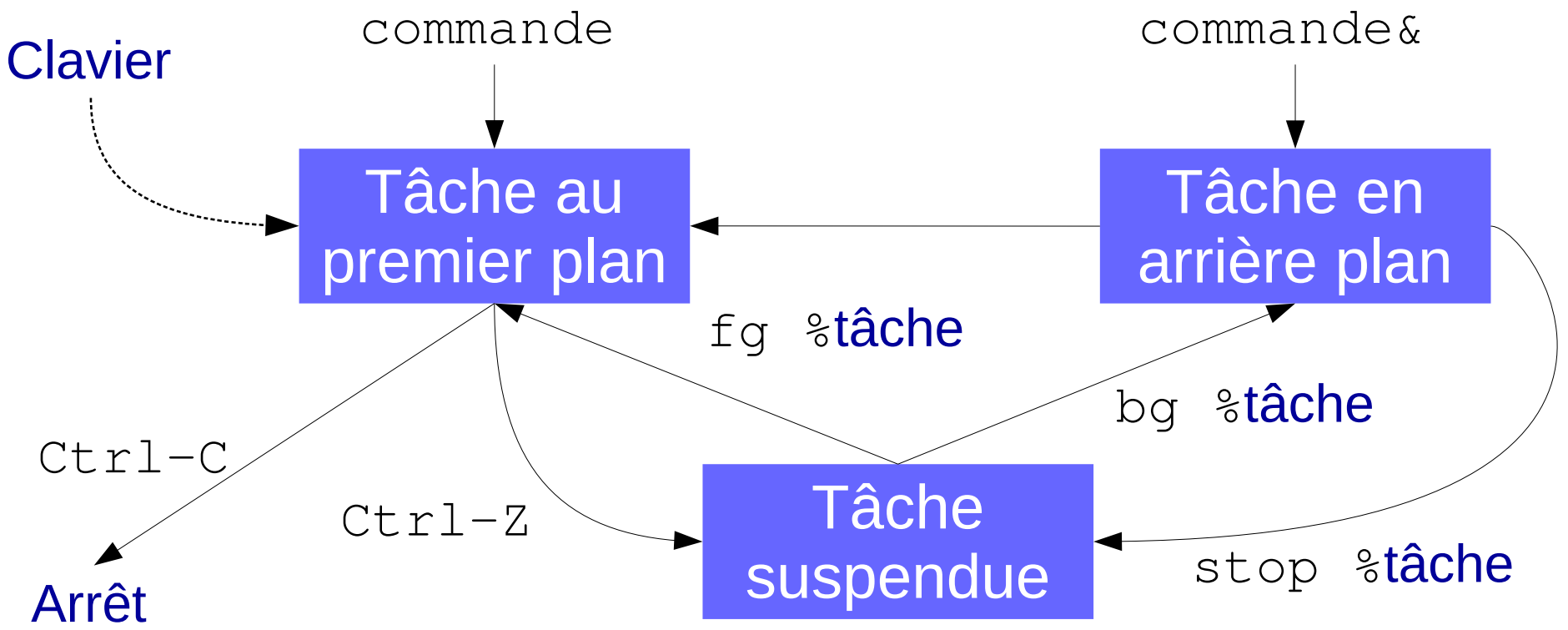
  peut être un fichier simple ou un répertoire
- option

  `-r` → récursif

Plan

- SE & gestionnaires de fenêtres
  - un tour d'horizon
- Gestion des fichiers
  - systèmes de fichiers, commandes & droits
- Gestion des processus
  - états et commandes
- Génération de fichiers
  - make, automake, autoconf, cmake, ...

Bash : contrôle des tâches

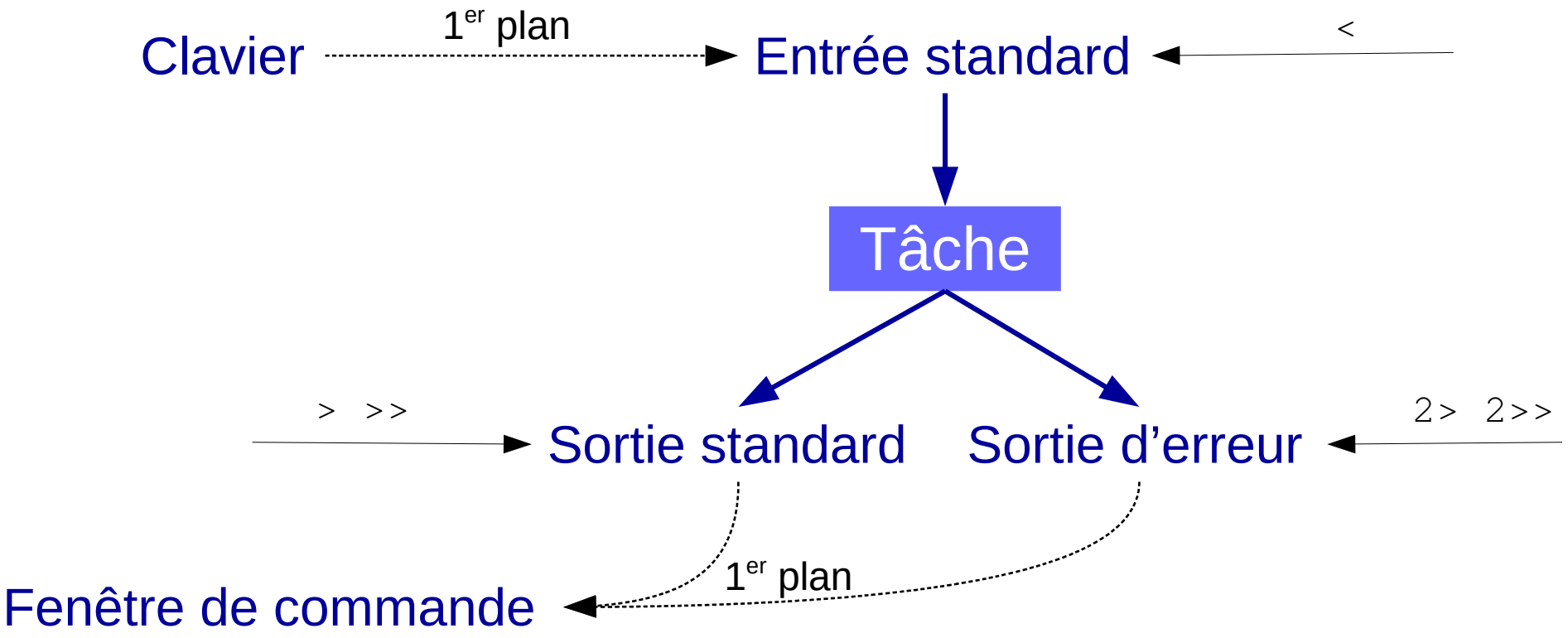


Tâches : surveillance et gestion

- Liste des tâches filles : `jobs`
- Liste des utilisateurs & charge : `w`
- Liste des tâches :
  - statique : `ps`
  - dynamique : `top`
- Arrêt d'un processus :

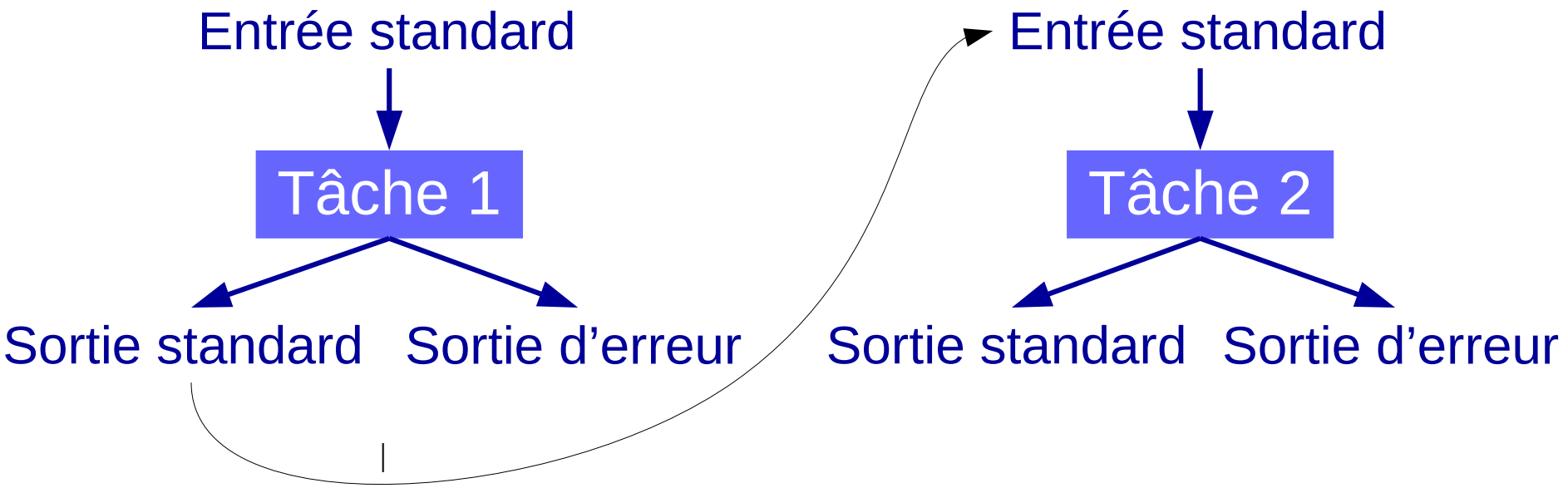
  `Ctrl-C` (1er plan), `kill` (tous)

Tâches : redirections



Simple (>) écrase, double (>>) concatène

Tâches : tunnel



Exemples : … `| less`, `ls /etc | wc -l`, …

Plan

- SE & gestionnaires de fenêtres
  - un tour d'horizon
- Gestion des fichiers
  - systèmes de fichiers, commandes & droits
- Gestion des processus
  - états et commandes
- Génération de fichiers
  - make, automake, autoconf, cmake, ...

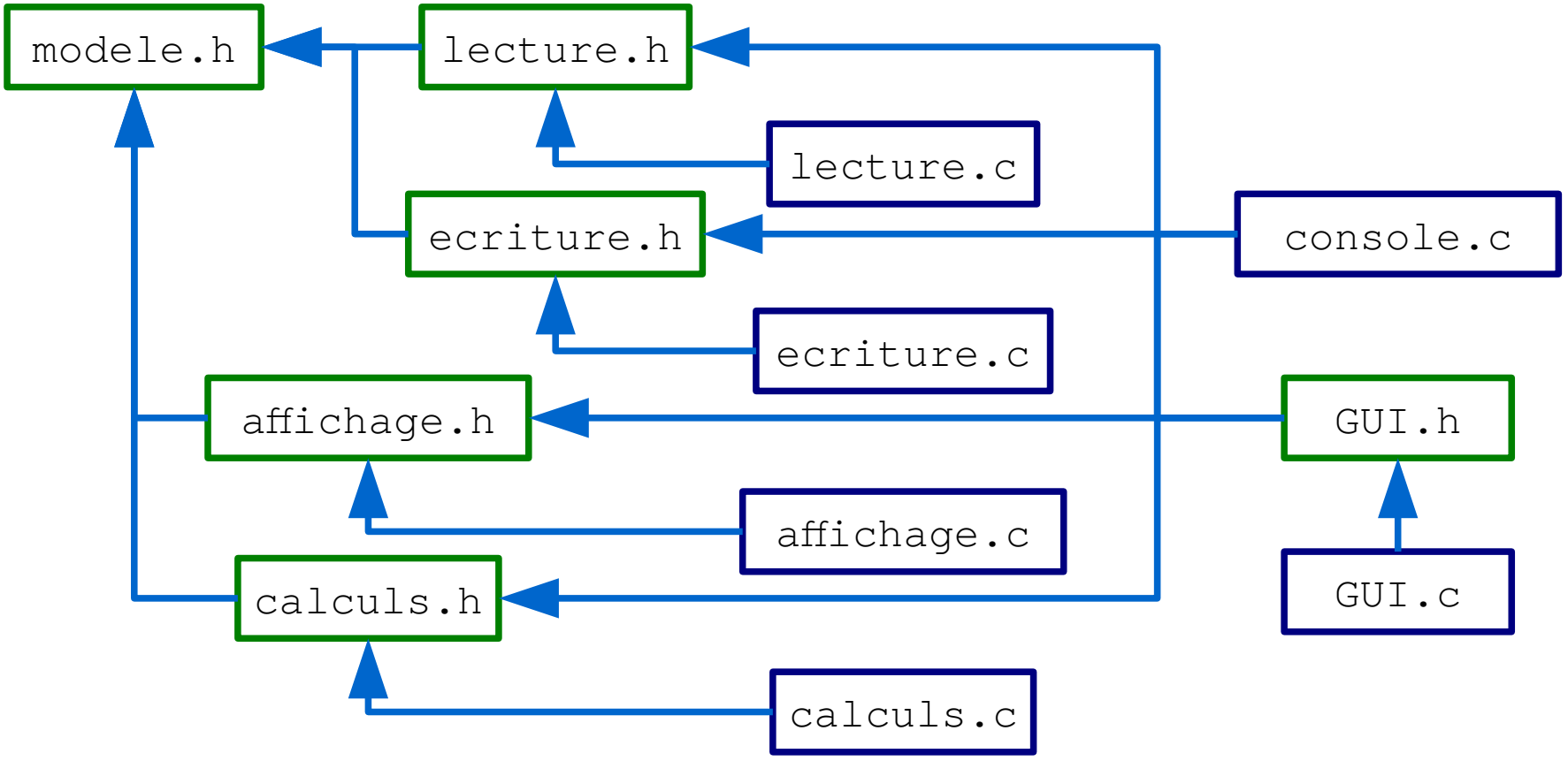
Makefile : intérêt

- Programme conséquent → code fragmenté
- Découpage selon thématique

  E/S dans fichier, affichage, calculs, …
- Compilations séparées & liaison finale (C/C++)
- Problème :

  gestion des dépendances & compilation minimale

Intérêt : exemple



Modification → Que doit-on recompiler ? Comment faire ?

Make : autre usage

Document structuré avec des images complexes

- Images générées à partir d'une base
- Formules ou textes insérés dans les images
- Texte structuré contenant les images

Makefile pour générer les images et le document

Réponses = Makefile

Fichier contenant un ensemble de règles avec

- Nom de la règle
- Condition(s) d'activation
- Commande(s) à utiliser

```
nom: dépendance(s)
     commande(s)
```

Makefile : exemple

```
all: console gui

console: lecture.o ecriture.o … console.c
	gcc -o console lecture.o … console.c
gui: lecture.o … GUI.h GUI.c
	gcc -o gui lecture.o … GUI.c
lecture.o: lecture.c lecture.h modele.h
	gcc -c lecture.c
…
```

Exemple compact

```
CC = gcc
FILES = lecture.o ecriture.o … calculs.o
all: console gui

console: $(FILES) console.c
	$(CC) -o $@ $^
gui: $(FILES) GUI.h GUI.c
	$(CC) -o $@ $^
%.o: %.c %.h modele.h
	$(CC) -c $<
```

Variables & règles prédéfinies

- $@ : la cible
- $< : la première dépendance
- $^ : toutes les dépendances
- $? : les dépendances récentes
- $* : le fichier sans extension
- Règles prédéfinies : `make -p (| less)`

Makefile : fonctionnement

- La commande `make` lance la première règle
  - souvent `all` (fait tout)
  - `make` *`regle`* lance la règle indiquée
- Pour chaque règle
  - vérifie les dépendances (…)
  - si nécessaire (…), lance la commande associée

Makefile : règle sans dépendance

- Nettoyage :

```
clean:
	rm $(FILES)
```

- Problème :

  pas d'activation si un fichier `clean` existe !

- Forcer le lancement de ces règles

  `.PHONY` force l'exécution des règles

```
.PHONY: clean
```

Makefile : règles / var. implicites

- Nombreuses règles définies

  compilation C, C++, Pascal, Fortran, …
- Nombreuses variables

  CC, CPPFLAGS, CFLAGS, CXX, CXXFLAGS…
- Inclus les variables de la fenêtre de commande

(cf GNU Make Manual)

Makefile : encore plus…

- Commandes silencieuses

  ajouter `@` avant la commande, ou utiliser l'option `-s`

- Gestion des dépendances

  génération par `cpp`, `makedepend` ou `gcc`

- Hiérarchie

  un makefile par répertoire + `include`

- Fonctions

  conditionnelles + fonctions de traitement

Makefile : références

- GNU Make Manual (en), une synthèse (fr)
- Tutoriels en ligne
  - préférez les sites universitaires / connus
  - recoupez les références entre elles
- Dépendances : simple ou complet (en)

Makefile : compléments

- Générer un makefile dépendant de l'installation
  - Trouver les outils : autoconf
  - Générer le makefile : automake

- Autres approches : Cmake, qmake, …