



Introduction aux Systèmes d'Exploitation

Présentation: Stéphane Lavirotte
Auteurs: ... et al*

(*) Cours réalisé grâce aux documents de :
Sacha Krakowiak, Stéphane Lavirotte, Jean-Paul Rigault,
Guillaume Salagnac

Mail: Stephane.Lavirotte@univ-cotedazur.fr

Web: <http://stephane.lavirotte.com/>

Université Côte d'Azur



Historique, Rôle et Fonctions d'un Système d'Exploitation

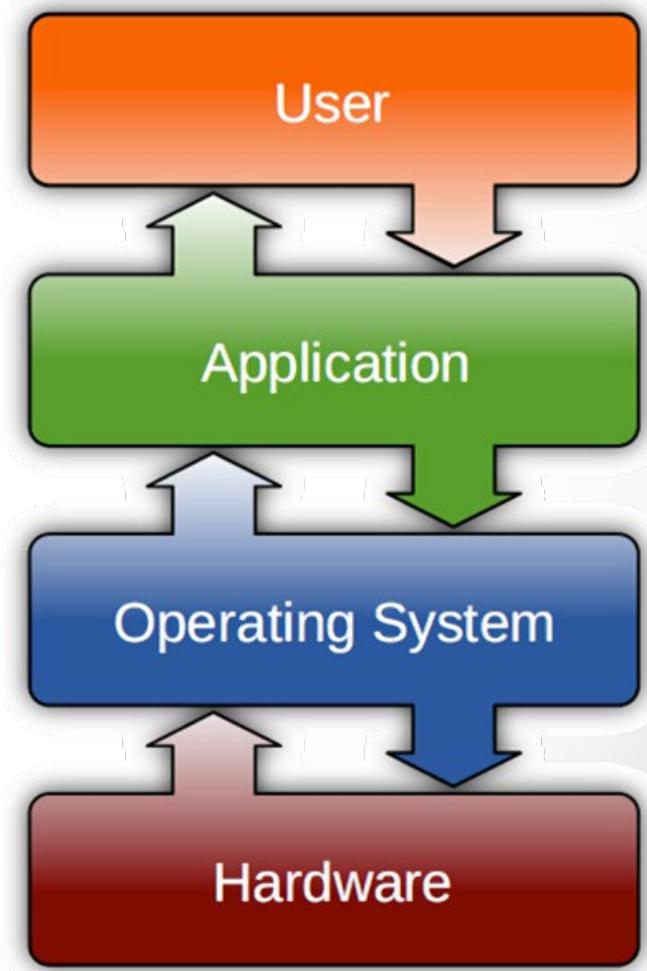
Introduction

Introduction

- ✓ « *Un système d'exploitation est une collection de choses qui ne tiennent pas dans un langage. Ça ne devrait pas exister.* »
 - Dan Ingalls, *Design Principles Behind Smalltalk*, Byte Magazine, August 1981
- ✓ Possibilité de mettre en œuvre directement sur un ordinateur via un langage
- ✓ Mais nombre de « services » ne sont pas du ressort d'un langage
 - Intendance commune à tous les langages
 - Ex: gestion de fichiers, affichage graphique, communication, entrées/sorties, sécurité, ...
 - Evite les redondances

Point de Vue : Utilisation des applications...

- ✓ **Un système d'exploitation**
 - Gère les ressources matérielles
 - Gère les besoins logiciels
 - Fournit les services communs aux programmes



Point de Vue: ... mais en réalité

✓ Utilisateur

- Humain devant la machine
 - Utilisateur final ou développeur
 - Interagit directement... avec le matériel !

✓ Applications

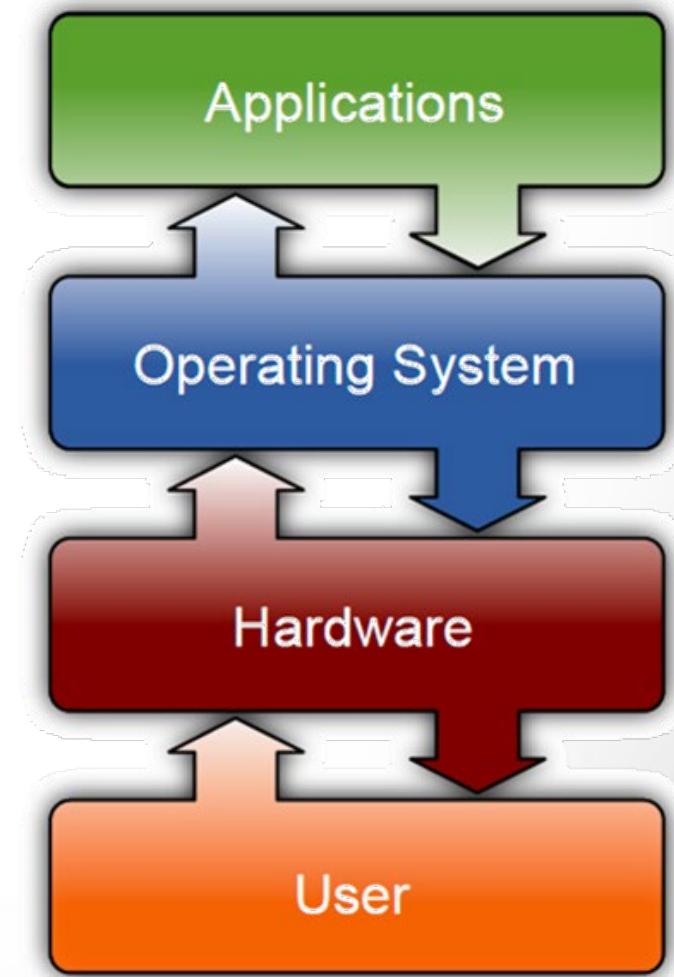
- Logiciels avec lesquels interagit l'utilisateur

✓ Matériel

- Machine physique et ses périphériques

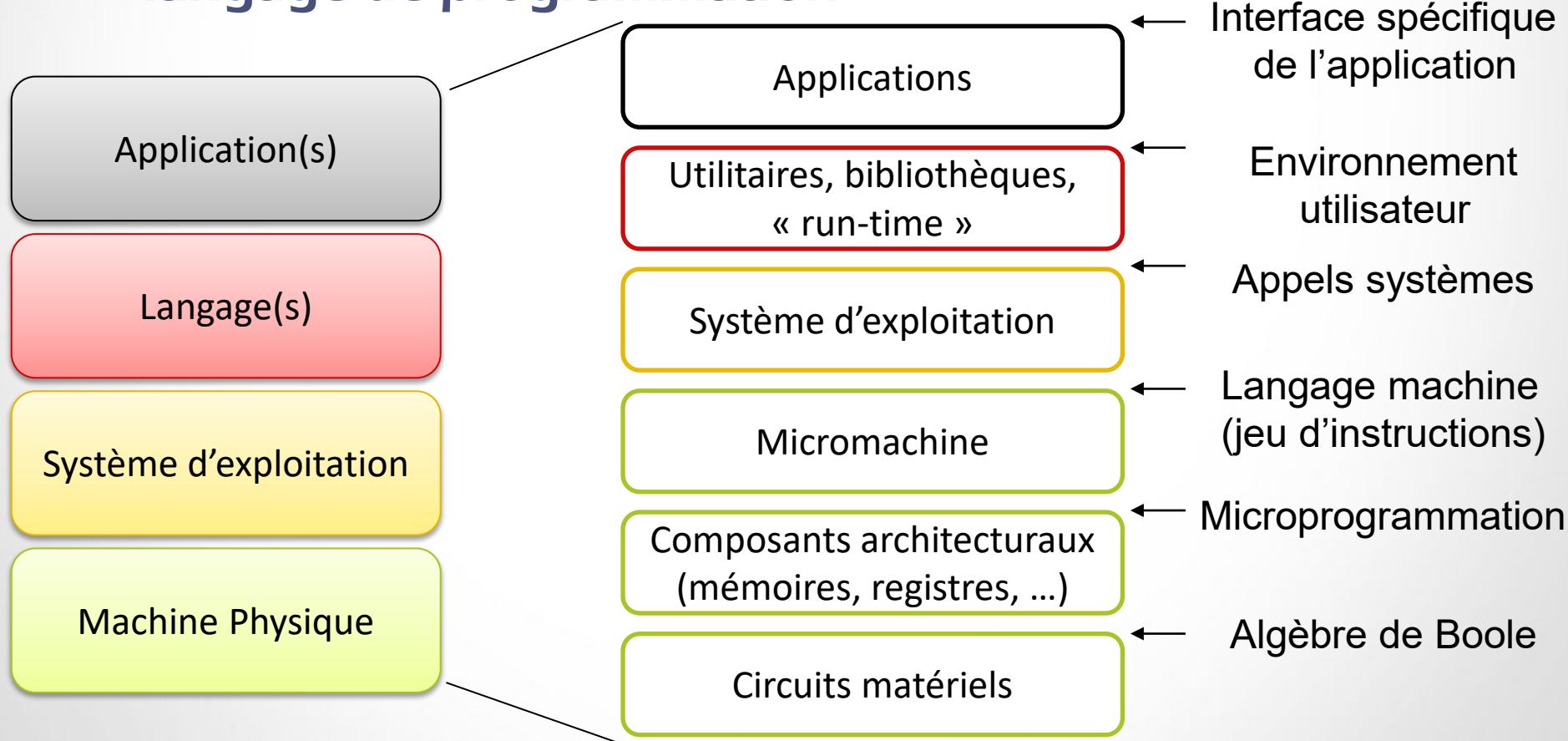
✓ Système d'Exploitation

- Tout le reste entre le matériel et les applications



Hiérarchie de Machines et de Langages (avec des interfaces)

- ✓ Combler l'écart entre la machine physique et un langage de programmation



La Réalité est plus Complexé

- ✓ **Nombre de couches**
 - Plus ou moins important
 - Ou pas de couche du tout (exécution directe sur la micromachine)
- ✓ **Séparation entre les couches**
 - Pas aussi nette certaines fois
- ✓ **La machine peut être partagée entre plusieurs entités**
 - Plusieurs utilisateurs
 - Plusieurs programmes (écrits avec divers langages de programmation)
- ✓ **Rôle du Système d'Exploitation**
 - Transformateur d'interface (cacher la complexité)
 - Fournir des services de base (IHM, stockage persistant, accès réseau, gestion du temps, ...)
 - Être le gérant des ressources

Fonctions et Rôle d'un Système d'Exploitation

✓ Gestion des ressources (partagées)

- Ressources physiques
 - Processeur(s), mémoire, disques, périphériques...
- Programmes, processus...
- Information
 - Désignation, localisation
 - Partage et échange entre usagers
- Protection, sécurité, confidentialité

✓ Fonctions diverses

- Statistiques, facturation...
- Gestion du temps, analyse de performances...

✓ Fournir une interface commode

Attribution des ressources
Prévention/Résolution des conflits

Classification des Systèmes d'Exploitation par Type d'Applications

- ✓ **Ordinateur individuel**
 - mono-utilisateur
 - mono-tâche (?)
 - peu de partage
 - peu de ressources
- ✓ **Commande « temps réel », système embarqué**
 - E/S riches
 - temps de réponse
 - parallélisme, concurrence
 - fiabilité, sûreté de fonctionnement
- ✓ **Système « temps partagé »**
 - multi-application, « universel »
 - nombreux services
 - protection inter-utilisateurs
 - temps de réponse « humain »
- ✓ **Systèmes transactionnels**
 - grande masse de données
 - partage, cohérence
 - nombreux utilisateurs
 - temps de réponse « humain »
- ✓ **Systèmes graphiques, CAO, images**
 - interactivité
 - puissance de calcul
 - grande masse de données

Classification des Systèmes d'Exploitation par Type d'Architecture de Système

✓ Système centralisé

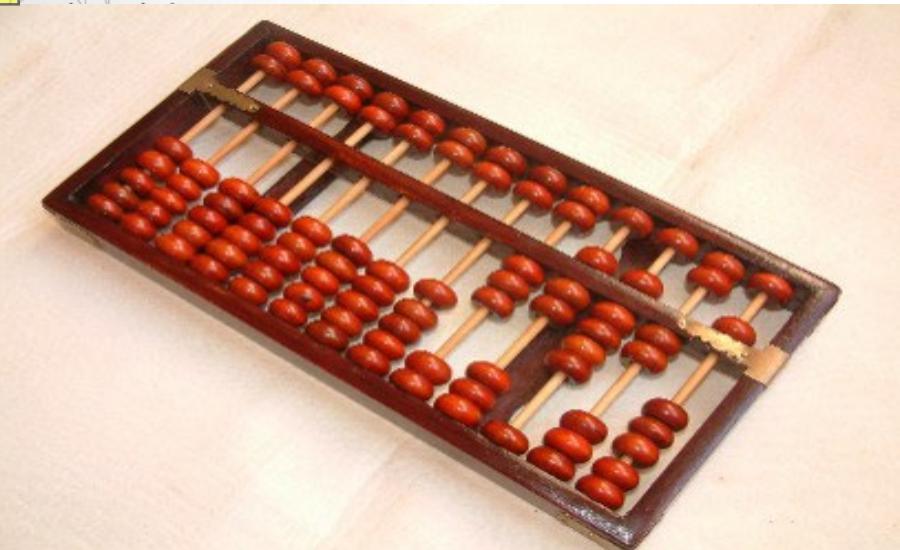
- mono/multi-utilisateur
- mono tâche/multitâche
- monoprocesseur/
multiprocesseur
asymétrique

✓ Système multiprocesseur

- processeurs
banalisés/dédiés
- activités migrantes/fixes
- mémoire partagée/locale

✓ Système réparti

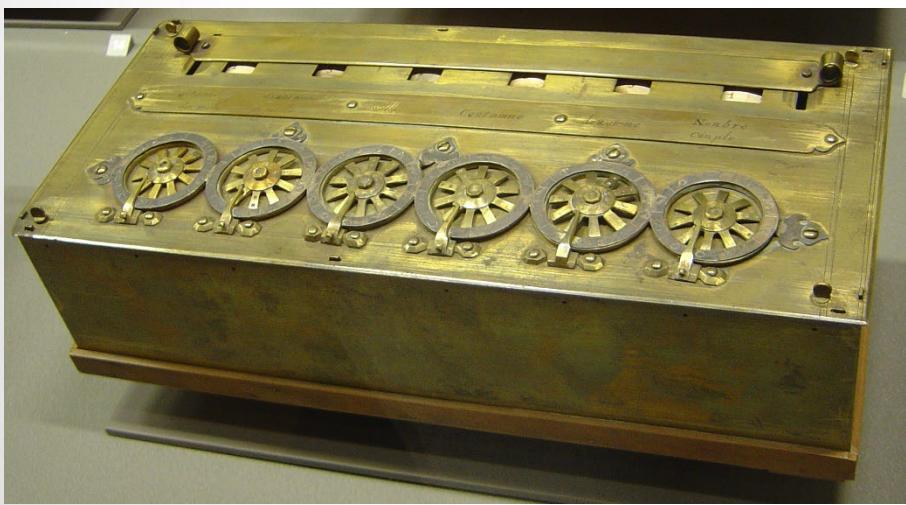
- processeurs
homogènes/hétérogènes
- systèmes d'exploitation
homogènes/hétérogènes
- données
locales/réparties/dupliquées
- activités migrantes/fixes
- localisation d'activités ou de
données transparentes ou
explicites



Boulier (ici un « suan pan » chinois)



Machine d'Anticythère



Calculateur mécanique (1623, 1642, 1673)



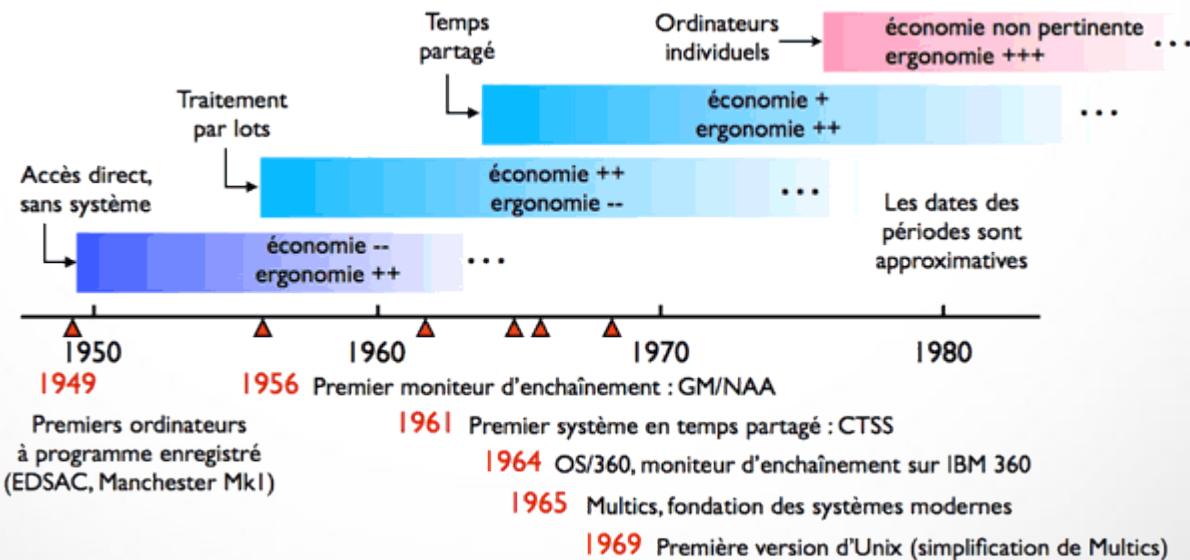
Premières machines programmables (1725)

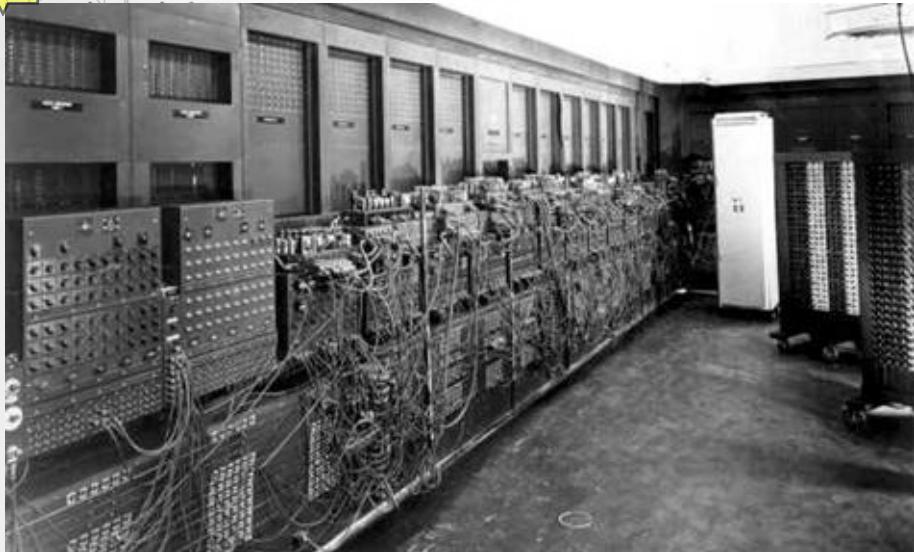


Chronologie Simplifiée des Premiers Systèmes

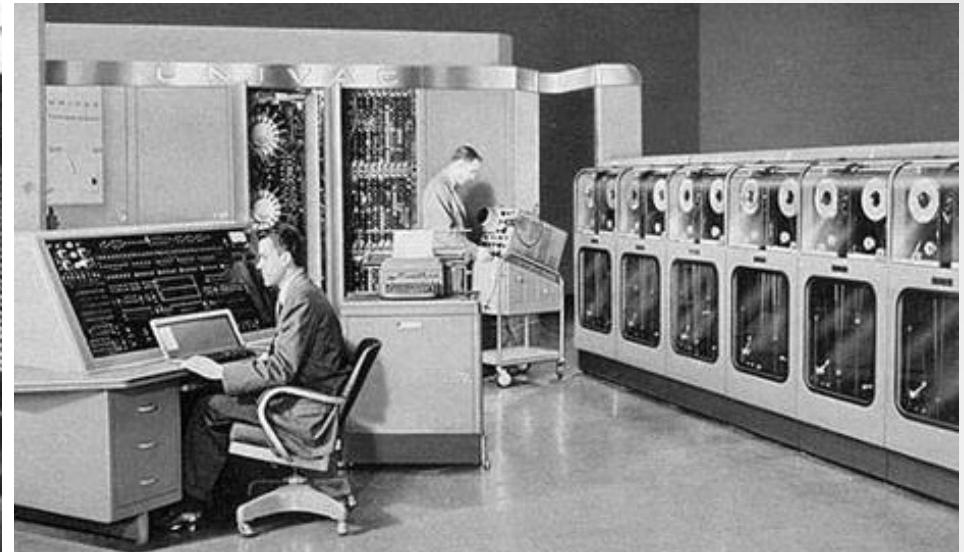
- ✓ La conception des systèmes d'exploitation est soumise à une tension entre deux objectifs contradictoires
 - Améliorer le confort des utilisateurs
 - On peut parler « d'ergonomie » (interface commode)
 - Exploiter efficacement les ressources physiques des machines
 - On peut parler « d'économie » (gestion des ressources)

✓ Histoire





Ordinateur entre 1945-1955



Ordinateur entre 1955-1962



Mini-Ordinateur entre 1960-1980

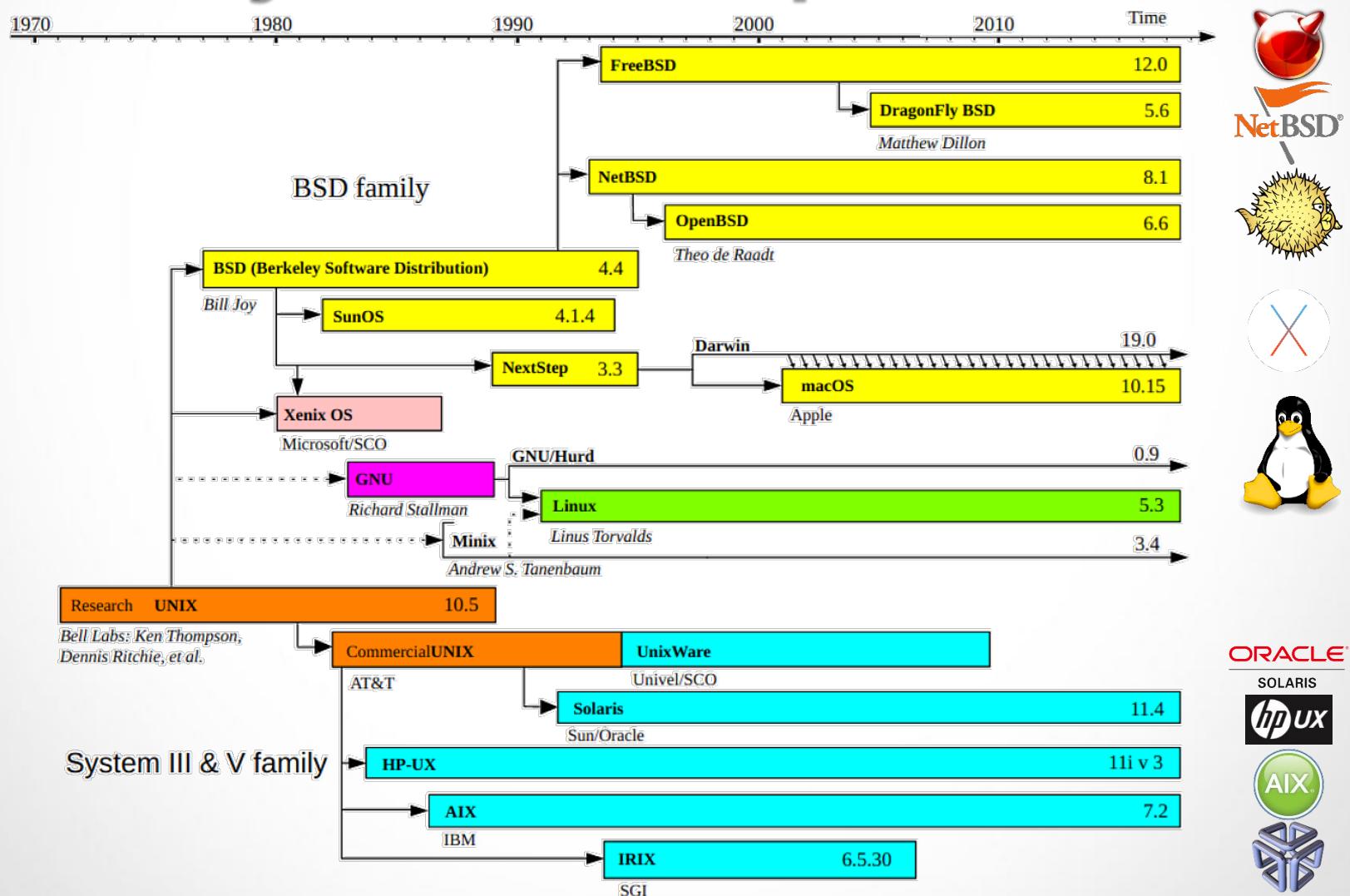


Micro-Ordinateur à partir de 1980

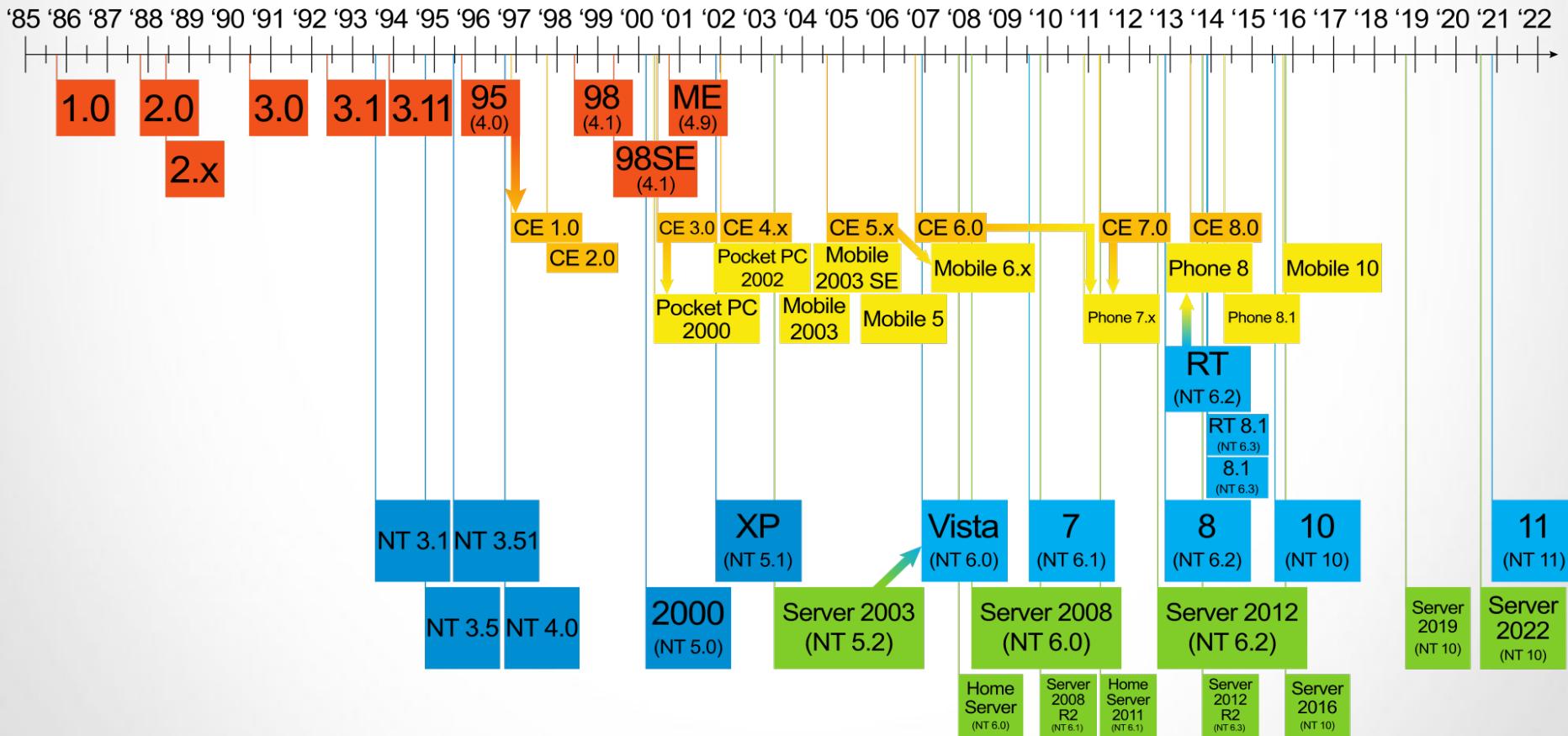
Brève Histoire des Systèmes d'Exploitation

	1945-1955	1955-1962	1960-1968	1965-?	1980-?
Technologie	Tubes à vide	Transistors	Circuits imprimés	MOS	VLSI Réseaux locaux
Langages	Langage-machine	Fortran, Cobol, Algol	Langages de haut niveau	Langages de haut niveau	Langages de haut niveau
Interaction	« Aux clés »	Cartes perforées, listing	« Batch », télétypes	Multi-utilisateurs, multitâches, temps partagé	Ecran bitmap, souris Stations de travail
Système	Pas de système d'exploitation à proprement parlé: Monomode Mono-tâche	Traitement par lots (« batch »)	Multiprogrammation Monomodes: Batch, temps réel, transactionnel, temps partagé	Intégrés multimodes Mémoire virtuelle Machine virtuelle Microprogrammation Gammes compatibles	Répartis Homogène ou hétérogène Partage et duplication d'information
Exemples	Whirlwind I UNIVAC I	IBM 701 (General Motors)	FMS , IBSYS (IBM) SAGE (temps réel) SABRE (transactionnel) ATLAS, CTSS, CMS , MCP , ... (temps partagé)	MCP (Burroughs 170) OS/360 (IBM) VM/370 (IBM) MULTICS (MIT) Unix (Bell Labs) VMS (DEC)	LOCAIS (UCB) UNIX/NFS (SUN) DECNET (DEC) Novell Windows NT

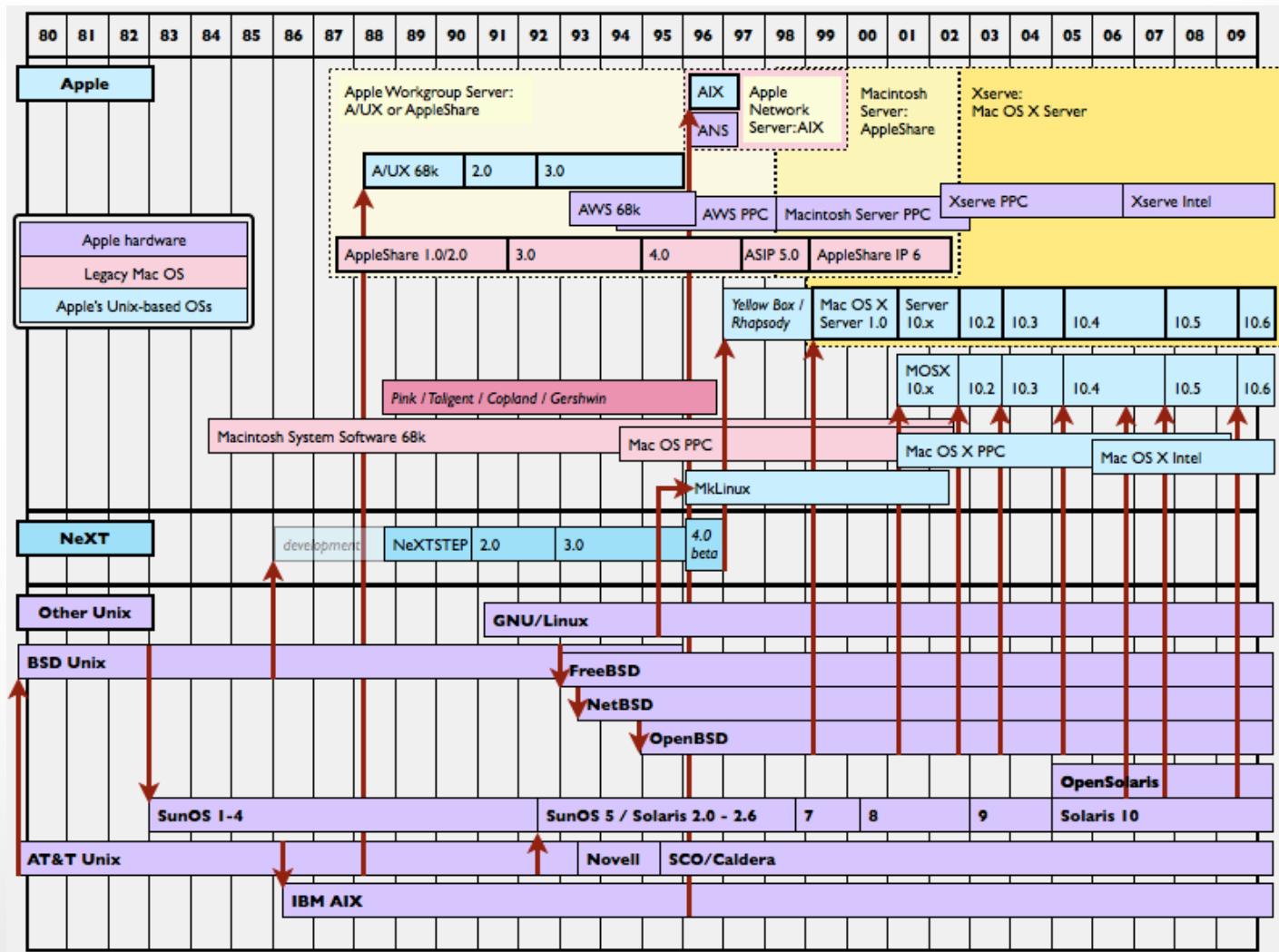
Frise Chronologique des Systèmes d'Exploitation Unix



Frise Chronologique des Systèmes d'Exploitation Windows



Frise Chronologique des Systèmes d'Exploitation Apple





Système d'Exploitation

Qu'est ce qu'un système d'exploitation ?

Caractéristiques d'un Système d'Exploitation

- ✓ **Un système d'exploitation est un ensemble de programmes qui dirige l'utilisation des capacités de la machine**
- ✓ **But:**
 - Faire une abstraction du matériel
 - Gérer le temps (temps partagé, temps réel)
 - Gérer la distribution (entre les processeurs, les mémoires, les périphériques) pour augmenter l'efficacité et abstraire
- ✓ **Fonctionnalités**
 - Servir les requêtes des processus
 - Appels systèmes: Read, Write, Open, ...
 - Traiter les exceptions matérielles dues aux processus
 - Déroulements: Division par 0, Débordement de pile, ...
 - Gérer les interruptions matérielles
 - Interruptions: clavier / souris, réseau, ...
 - Fournir un ensemble de services spécifiques
 - Assurer des tâches d'entretien du système (swap, caches, pages, ...)

Eléments d'un Système d'Exploitation

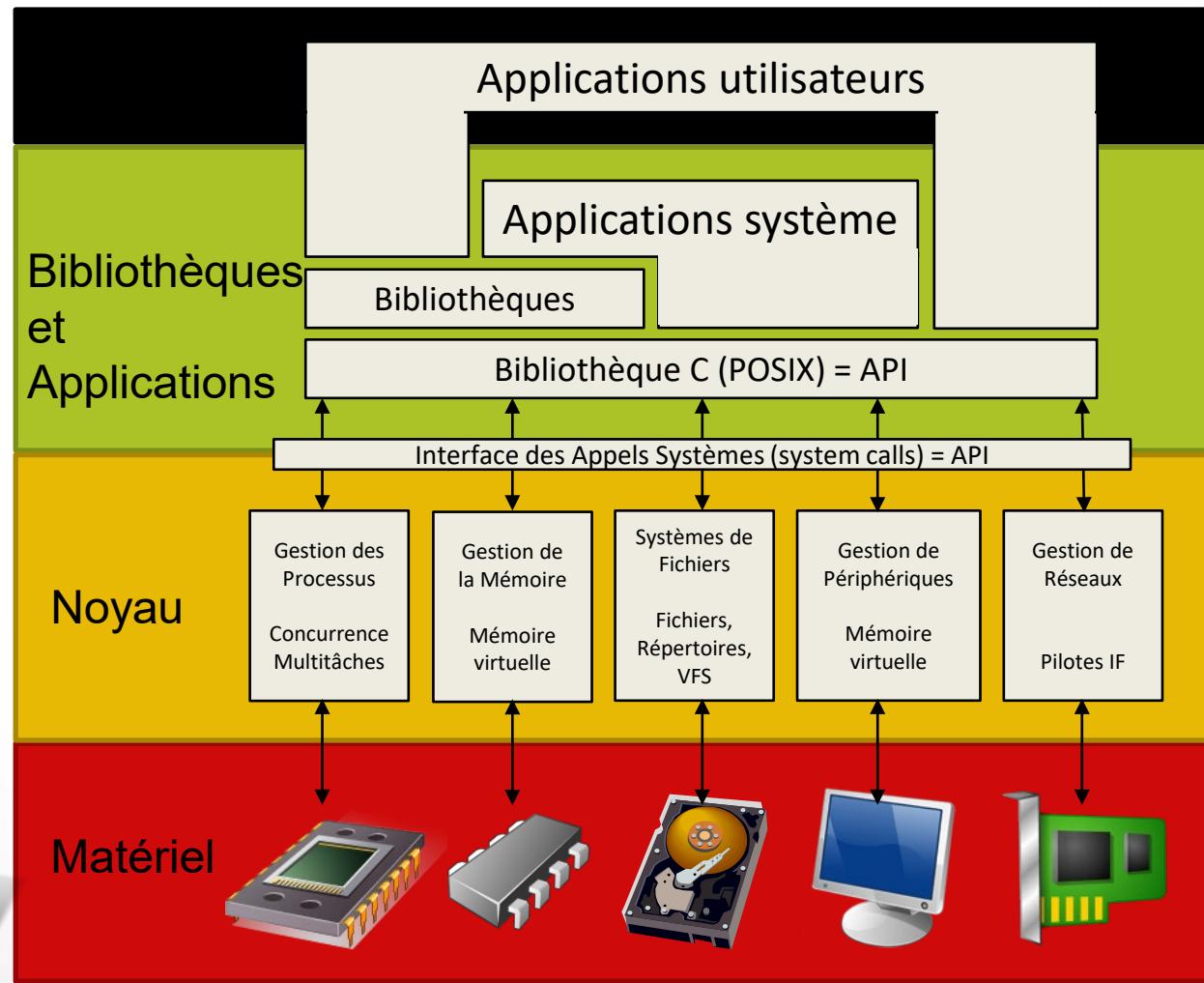
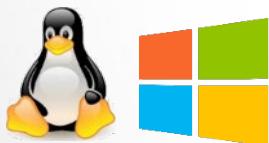
✓ 3 éléments principaux

- Les programmes utilitaires
 - Ensemble de programmes outils permettant de manipuler le système à base de commandes « basiques »
 - Ex: shell, cp, rm, mount, ...
 - Voir cours PeIP 1: « [Environnement Informatique 1](#) »
- La/Les librairies
 - Bibliothèque(s) standardisées de fonctionnalités pour les programmes utilisateurs
 - Librairie C (libc), librairie math (libm)
 - Ce cours SI3: « [Programmation Systèmes](#) »
- Le Noyau
 - Programme qui est le premier à s'exécuter après le chargeur
 - Fournit les abstractions pour la gestion des processus de la mémoire, des systèmes de fichiers, ...
 - Voir cours SI5: « [Systèmes d'Exploitation avancés](#) »

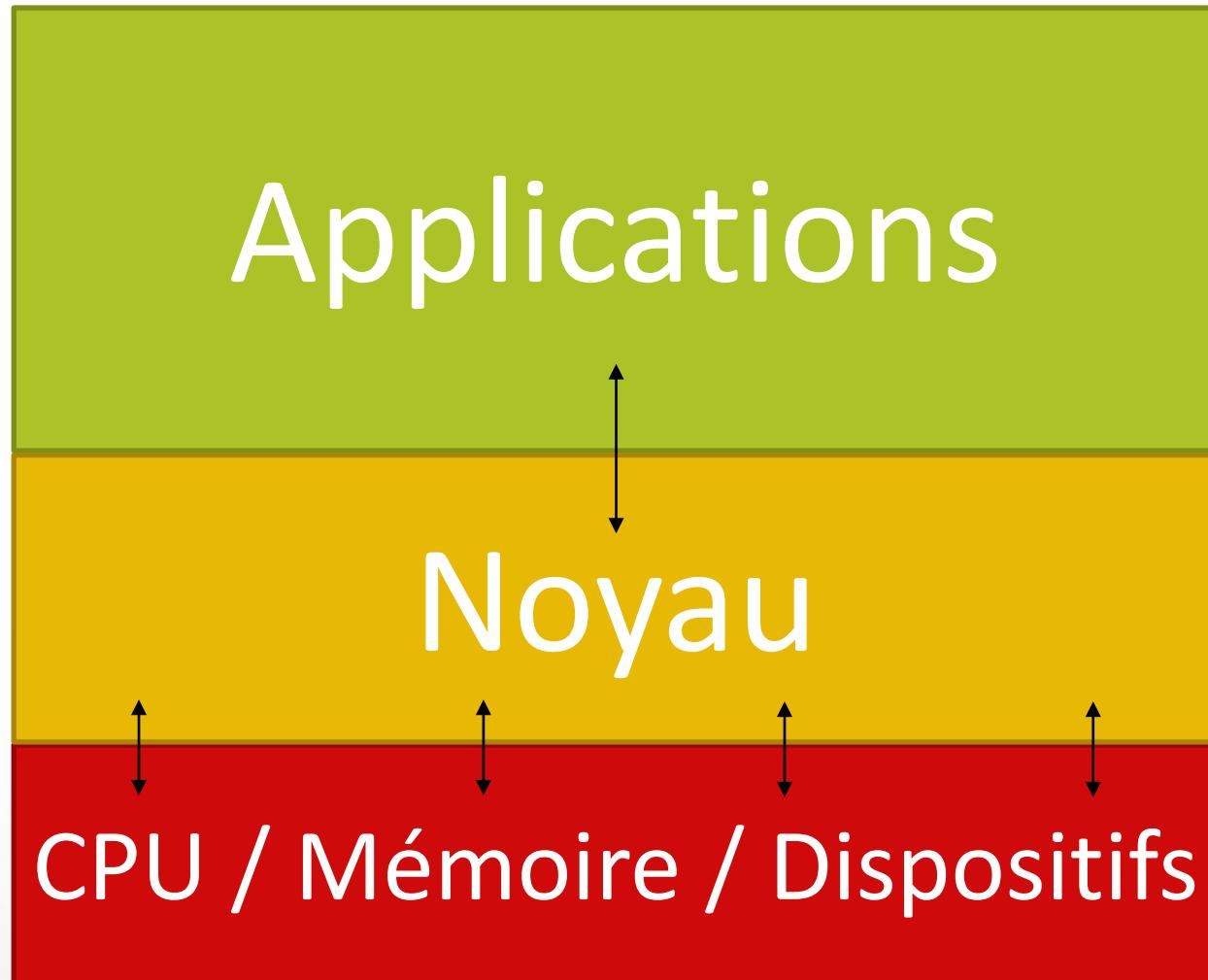
Architecture générale d'un Système d'Exploitation



XNU



Architecture générale d'un Système d'Exploitation



En des termes simples

- ✓ **Le Système d'Exploitation est là pour**
 - Bien partager les ressources
- ✓ **Exemples**
 - Envoyer l'appui d'une touche à la bonne application
 - S'assurer que les processus n'ont pas accès à la mémoire des autres
- ✓ **Gestion des ressources**
 - Partage de la ressource de calcul
 - Gestion des processus
 - Partage de la mémoire entre les différents processus
 - Gestion mémoire
 - Partage des dispositifs (clavier, souris, écran, ...)
 - Gestion des entrées/sorties

En résumé

un Système d'Exploitation

✓ **Trois grandes fonctionnalités:**

- Gérer la mémoire
- Gérer les entrées-sorties
- Gérer les processus

✓ **Qualités attendues:**

- Fiabilité, robustesse
- Prédictibilité, déterminisme
- Ergonomie, facilité d'utilisation
- Efficacité, performances
- Généralité, universalité
- Flexibilité, adaptabilité, extensibilité
- Transparence

Cacher la Complexité sous une « jolie interface »

- ✓ Et c'est vraiment la formule utilisée dans les livres et surtout le livre de référence:
 - Tanenbaum. Modern Operating Systems (4th ed, 2014). page 5

SEC. 1.1

WHAT IS AN OPERATING SYSTEM?

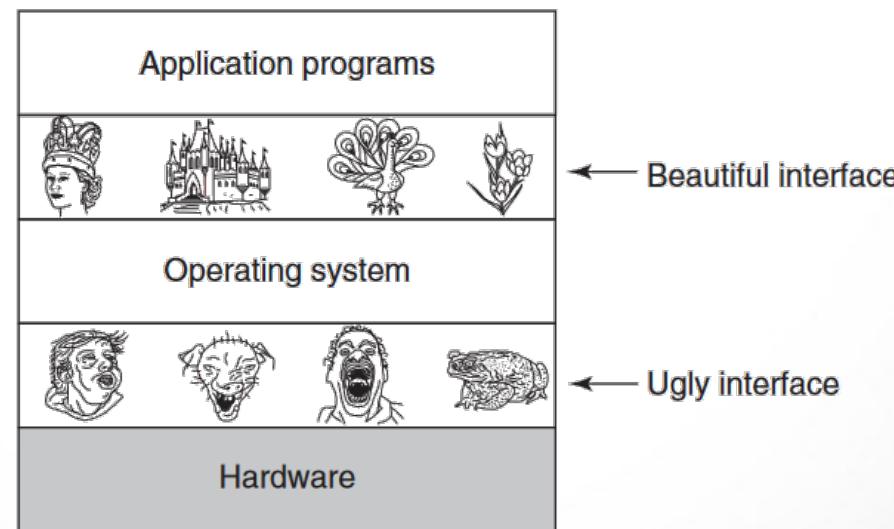


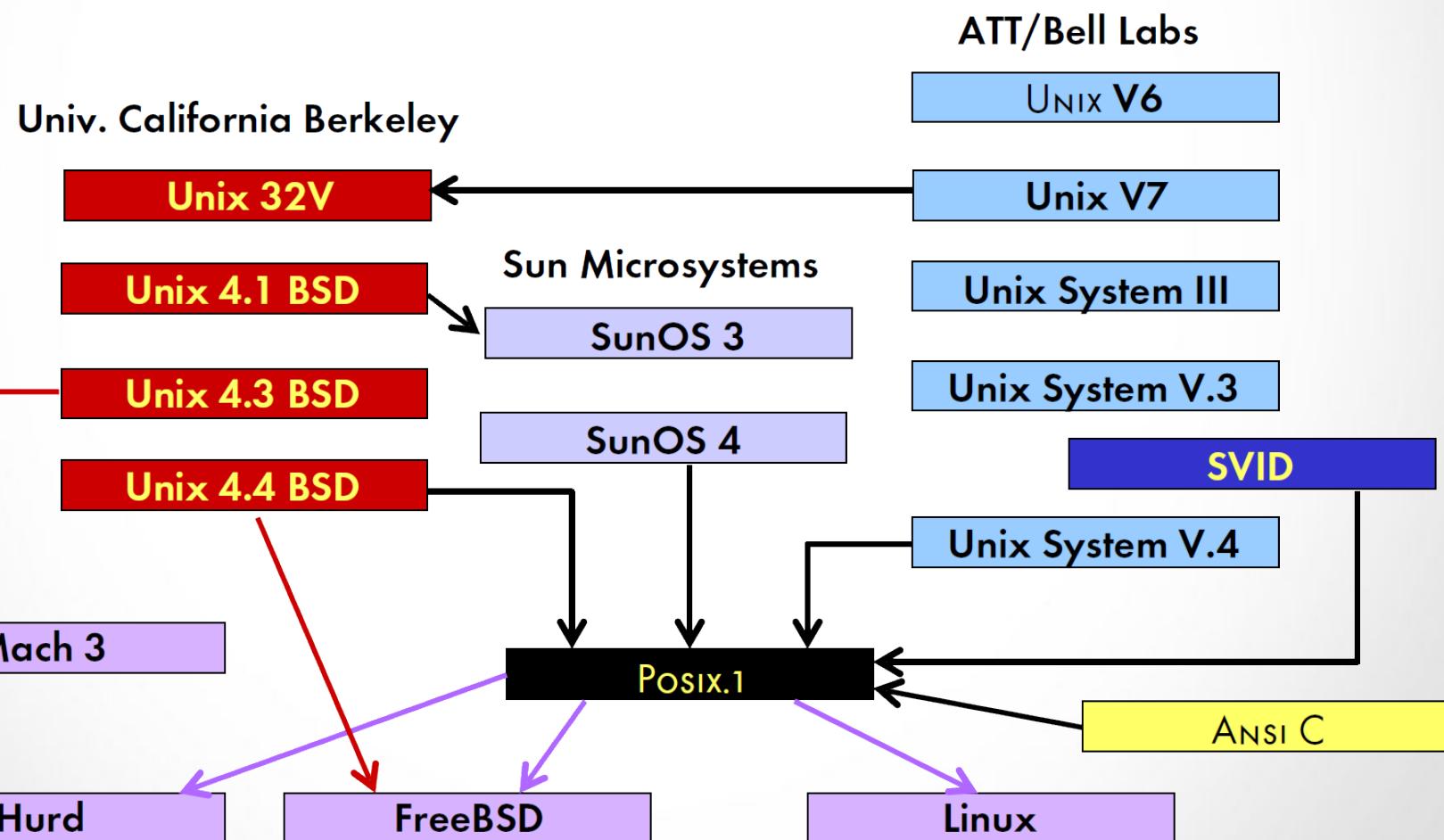
Figure 1-2. Operating systems turn ugly hardware into beautiful abstractions.



Une Interface Normalisée

Posix.1

D'Unix à Posix



Genèse de Posix

✓ Motivations

- API : Interface portable de programmation d'applications
- Portabilité au niveau source
- Compatibilité avec Ansi C
- Unification des versions d'Unix et ouverture à d'autres systèmes d'exploitation

✓ Déroulement des travaux

- POSIX est une famille de normes techniques définie depuis 1988
- *Posix = Portable Operating System Interface*, et X vient de Unix
- Groupes de travail : /usr/group, IEEE 1003, Ansi
- Normes : ISO 9945-1 (1990) ; révisions en 1996, 2001

Normes Posix

✓ Interface portable de programmation

- Posix.1, 1a/1b (extensions temps-réel), 1c (threads)
- Shell et commandes de base (Posix.2)

✓ Supports divers

- Réseaux
- Langages (Fortran, Ada...)
- Systèmes (bases de données, transactionnels...)
- etc.

Compatibilité Posix. 1-2001

✓ Posix et Unix

- Posix est fondé sur Unix BSD et Unix SVR4
- La plupart des systèmes Unix actuels proposent la compatibilité Posix

✓ Posix et les systèmes non-Unix

- Bibliothèques d'émulation (e.g. Cygwin)

✓ Posix et Ansi C

- Description première de l'API en Ansi C
- Support d'autres langages prévu
- L'API Posix est un sur-ensemble de la bibliothèque standard d'Ansi C

L'API Posix.1-2001

✓ Ensemble de fonctions

- appels-système ou bibliothèque
- fichiers d'entête
 - <assert.h> <dirent.h> <ctype.h> ...
 - <sys/times.h> <sys/types.h> ...

✓ Compatibilité stricte

- gcc -ansi -pedantic -D_POSIX_SOURCE=1 ...

✓ Valeur de retour en cas d'erreur

- si int : -1
- si pointeur : 0 (NULL)

✓ Code de l'erreur (EACCESS, E2BIG...)

- extern int errno;

✓ Messages d'erreur

```
#include <stdio.h>
void perror(const char *my_message);
```

```
#include <string.h>
char *strerror(int errnum);
```

Et par la suite...

- ✓ **Nous explorerons la norme Posix pour**
 - La gestion des processus
 - La gestion des entrées-sorties
 - La gestion des signaux
 - La gestion de la mémoire
- ✓ **Nous comparerons Posix et Win32**
 - Pour voir les similarités des approches
 - Pour voir les différences des approches
- ✓ **Nous comparerons aussi Posix et ANSI C**