NMV :
Histoire et architecture du noyau Linux
Version 15.01

Julien Sopena[1]

[1]julien.sopena@lip6.fr
Équipe REGAL - INRIA Rocquencourt
LIP6 - Université Pierre et Marie Curie

Master SAR 2ème année - NMV - 2016/2017

Grandes lignes du cours

Taxinomie des noyaux
- Noyaux monolithiques
- Micro-noyaux
- Noyaux modulaires monolithiques
- Micro-noyaux hybride

Histoire de Linux
- Histoire des systèmes Unix
- Linux

Architecture du noyau Linux

Structuration des sources du noyau

Outline

Taxinomie des noyaux

- Noyaux monolithiques
- Micro-noyaux
- Noyaux modulaires monolithiques
- Micro-noyaux hybride

Histoire de Linux

Architecture du noyau Linux

Structuration des sources du noyau

Les différents types de noyaux

On distingue généralement 4 grands types de noyaux :

- Les noyaux monolithiques,
- Les micro-noyaux,
- Les noyaux monolithiques modulaires,
- Les exo-noyaux.

Outline

Taxinomie des noyaux

- Noyaux monolithiques
- Micro-noyaux
- Noyaux modulaires monolithiques
- Micro-noyaux hybride

Histoire de Linux

Architecture du noyau Linux

Structuration des sources du noyau

Noyaux monolithiques : Définition

Définition

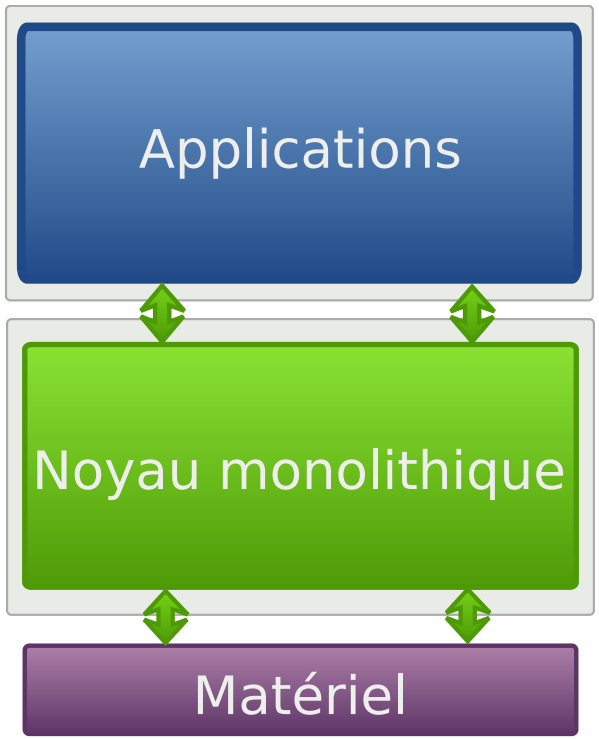
*Un **noyau monolithique** est un système d'exploitation dont l'ensemble des fonctionnalités du système et des pilotes sont regroupés dans un seul bloc de code et un seul bloc binaire généré à la compilation.*

Exemple

Premières versions de Linux, certains BSD ou certains anciens Unix.
Mais aussi des systèmes embarqués critiques : IOS de Cisco

L'aspect monolithique ne concerne pas forcement les sources qui peuvent être architecturées en bibliothèques bien distinctes les unes des autres. Cependant ce code très intégré est aussi compliqué à concevoir qu'à développer correctement.

Noyaux monolithiques : Schéma



Noyaux monolithiques : Caractéristiques

- S'exécute entièrement en *kernel space* $\Rightarrow \begin{cases} + \text{ Performances} \\ \text{-/+ Sécurité} \end{cases}$
- Code optimisé pour une architecture $\Longrightarrow \begin{cases} + \text{ Performances} \\ \text{- Portabilité} \end{cases}$
- Chargé entièrement en mémoire $\Rightarrow$ Problème de place mémoire

  $\Rightarrow$ Limite les fonctionnalités

Outline

Taxinomie des noyaux

- Noyaux monolithiques
- Micro-noyaux
- Noyaux modulaires monolithiques
- Micro-noyaux hybride

Histoire de Linux

Architecture du noyau Linux

Structuration des sources du noyau

Micro-noyaux : Définitions

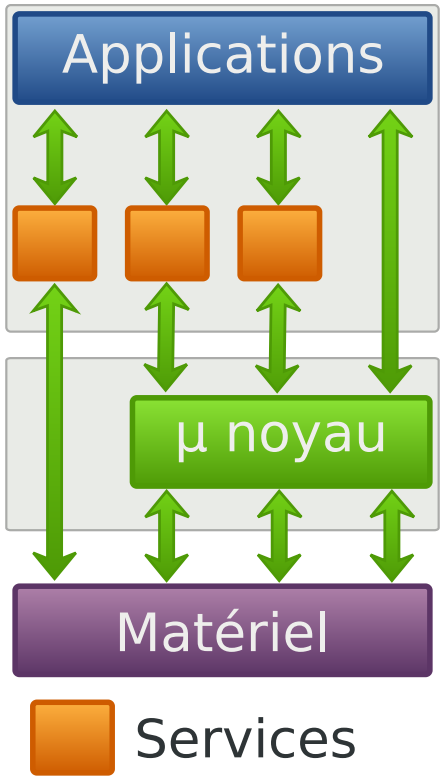
Définition

*Un **micro-noyau**, est le cœur d'un système d'exploitation cherchant à minimiser les fonctionnalités intégrées au noyau en plaçant la plus grande partie des services du système d'exploitation à l'extérieur de ce noyau, c'est-à-dire dans l'espace utilisateur. Ces fonctionnalités sont alors fournies par de petits serveurs indépendants possédant souvent leur propre espace d'adressage.*

Définition

*Un **micro-noyau enrichi** est l'ensemble logiciel formé par le micro-noyau et les services déplacés en espace utilisateur.*

Micro-noyaux : Schéma



Micro-noyaux : Caractéristiques

- Encombrement mémoire très faible $\Rightarrow$ Systèmes embarqués
- Services bas-niveau hors-noyau $\Rightarrow \begin{cases} + \text{ Sécurité renforcée} \\ + \text{ Souplesse} \\ - \text{ Performances limitées} \end{cases}$
- Interaction micro-noyau/services par une interface standard
$\Rightarrow \begin{cases} + \text{ Portabilité très importante} \\ - \text{ Bride les capacités et la diversité} \end{cases}$

Outline

Taxinomie des noyaux

- Noyaux monolithiques
- Micro-noyaux
- Noyaux modulaires monolithiques
- Micro-noyaux hybride

Histoire de Linux

Architecture du noyau Linux

Structuration des sources du noyau

Noyaux modulaires monolithiques : Définition

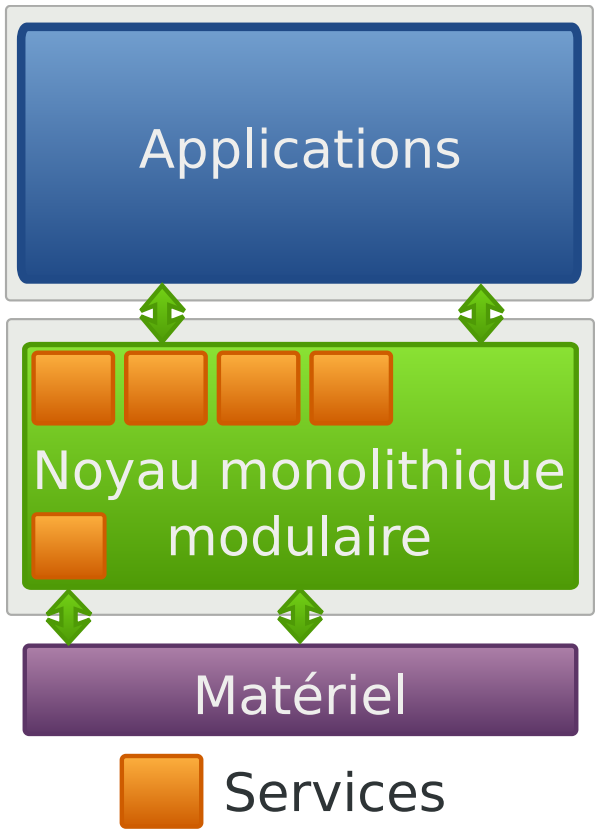
Définition

*Un **noyau monolithique modulaire** est un système d'exploitation dont les parties fondamentales du système sont regroupées dans un bloc de code unique (monolithique). Les autres fonctions, comme les pilotes matériels, sont regroupées en différents modules qui peuvent être séparés tant du point de vue du code que du point de vue binaire. Mais une fois chargées, ces fonctions s'exécutent en kernel space et partagent le même espace d'adressage que le coeur du noyau.*

Exemple

Linux, la plupart des BSD ou encore Solaris.

Noyaux modulaires monolithiques : Schéma



Noyaux modulaires monolithiques : Caractéristiques

- Interaction direct avec le matériel $\Longrightarrow \begin{cases} + \text{ Performances} \\ - \text{ Portabilité} \end{cases}$
- Code optimisé pour une architecture $\Longrightarrow \begin{cases} + \text{ Performances} \\ - \text{ Portabilité} \end{cases}$
- Services sous forme de module $\Longrightarrow \begin{cases} + \text{ Extensible} \\ + \text{ Portabilité} \end{cases}$
- Modules exécutés en *kernel space* $\Longrightarrow \begin{cases} + \text{ Performances} \\ - \textbf{ Sécurité} \end{cases}$

Outline

Taxinomie des noyaux

- Noyaux monolithiques
- Micro-noyaux
- Noyaux modulaires monolithiques
- Micro-noyaux hybride

Histoire de Linux

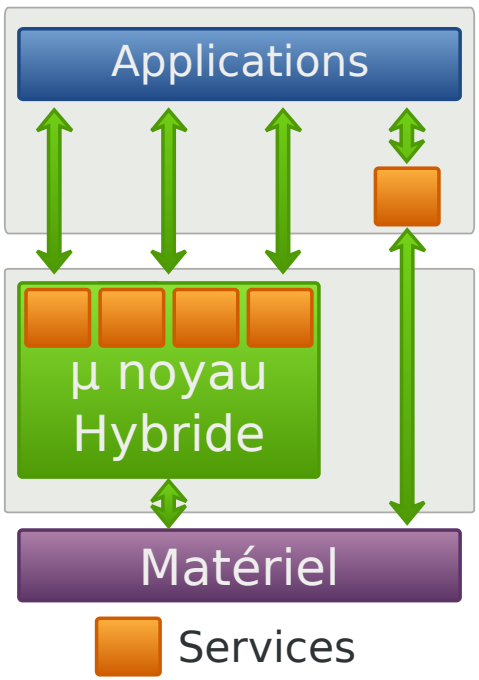
Architecture du noyau Linux

Structuration des sources du noyau

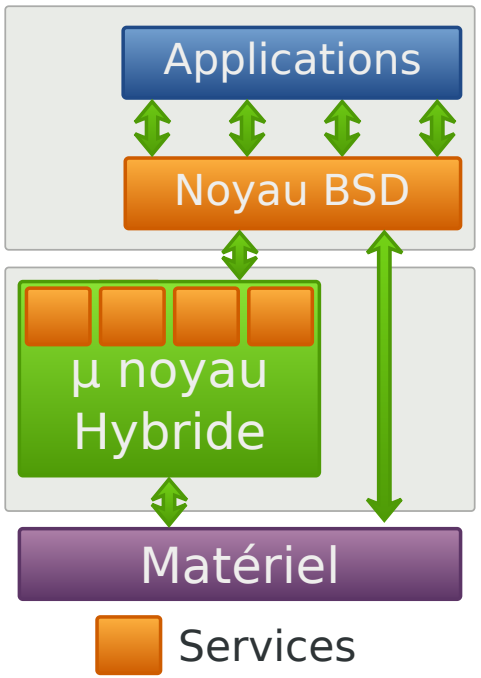
Micro-noyaux hybride : Définitions

Définition

*Un **micro-noyau hybride**, est un micro noyau qui intègre directement sous forme de module certains services cruciaux, le reste des services restants à l'extérieur du noyau.*



Micro-noyaux hybride.



Exemple de Mach/XNU
le "micro-noyau" d'apple".

Exo-noyaux : définition

Définition

*Un **exo-noyau**, est un noyau minimal qui effectue uniquement un multiplexage sécurisé des ressources matérielles disponibles. Les applications peuvent sélectionner et utiliser des librairies formant le reste système d'exploitation ou implémenter les leurs.*

Exemple

Concept né du projet Aegis du MIT.

Si les *micro-noyaux* remonte le matériel au niveau d'une abstraction standard (*HAL*), les *exo-noyaux* descendent les applications à un niveau standard du matériel.

Exo-noyaux : Caractéristiques

- Interaction direct avec le matériel $\Longrightarrow \begin{cases} + \text{ Performances} \\ - \text{ Portabilité} \end{cases}$
- L'ensemble des services en *user mode* $\Longrightarrow$ Sécurité très importante.
- Gestion de ressources physiques au niveau applicatif

$$\Rightarrow \begin{cases} + \text{ Limite les contraintes} \\ + \text{ Permet des applications spécifiques} \\ - \text{ Demande un nouveau type de programmation} \end{cases}$$

Outline

Taxinomie des noyaux

Histoire de Linux

- Histoire des systèmes Unix
- Linux

Architecture du noyau Linux

Structuration des sources du noyau

Outline

Taxinomie des noyaux

Histoire de Linux

- Histoire des systèmes Unix
- Linux

Architecture du noyau Linux

Structuration des sources du noyau

Origine du nom Unix

Mais d'où vient le nom Unix ?

Origine du nom Unix

Mais d'où vient le nom Unix ?

Le nom **Unix** fait référence au système d'exploitation **Multics** (*Multiplexed Information and Computing Service*) issu d'une collaboration entre le *MIT*, *Bell* et *General Electric*.

C'est en 1970, lorsque *Bell* quitte le projet, que Ken Thompson crée se que l'on nommera **Unix**.

À ses débuts le système **Unix** s'écrivait **Unics** :

Version 1 : pour *UNiplexed Information and Computing System*.

Version 2 : prononcer *"Eunuchs"* comme un *"Castrated Multics"*

Histoire des systèmes Unix

1969 Ken Thompson (*ATT Bell labs*) écrit la première version pour un PDP-7 de ce qui sera **Unix**

1970 Ken Thompson et Dennis Ritchie l'adaptent au DEC PDP-11/20.
Au passage ils développent le premier compilateur C.

1971 Première version d'**Unix**, avec l'assembleur du PDP-11/20.

1973 Ritchie et Thompson réécrivent le noyau d'Unix en C, c'est l'**Unix V4**.

1974 Les sources d'**Unix V5** sont distribuées librement aux universités.

1978 Sortie d'**Unix V7** mais *ATT* annonce une licence payante pour l'accès aux sources.

1979 *ATT* annonce son intention de créer une **version commerciale d'Unix**.
En réaction l'université de Californie à Berkeley lance sa propre variante : **BSD UNIX**.

1983 *ATT* met en vente la version commerciale du **System V**.
Sortie d'une des versions majeurs de **BSD** la **4.2**.

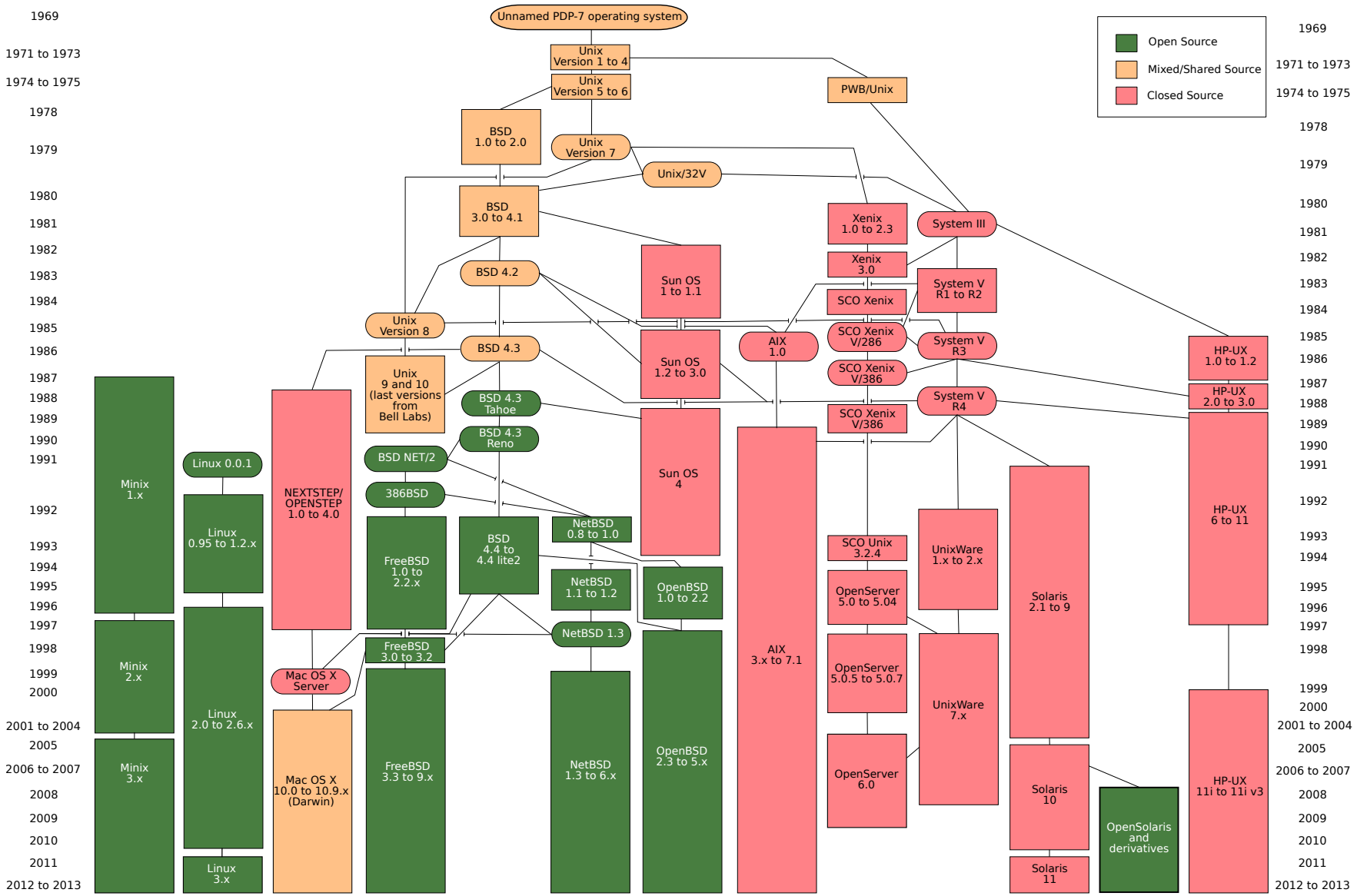
1987 *ATT* commercialise la version **System V R3**.
Sortie d'une autre version majeurs de **BSD** la **4.3**.
C'est cette année là qu'*ATT* et *Sun* choisissent d'unifier le **Système V**, **SunOS** et **BSD**.

1990 La coopération entre *Sun* et *ATT* donne naissance à la **System V R4** qui comporte de nouveaux standard d'unification d'UNIX.
En réaction, les autres grands constructeurs, dont DEC, HP et IBM, décident de créer l'**OSF (Open Software Foundation)**.

1991 La guerre des clones commence avec l'arrivée de **Linux** et **FreeBSD**.

1992 Sun développe **Solaris** un dérivé du Système V R4 avec la gestion des threads.

Histoire des systèmes Unix



Outline

Taxinomie des noyaux

Histoire de Linux

- Histoire des systèmes Unix
- Linux

Architecture du noyau Linux

Structuration des sources du noyau

Un petit message sur le forum Usenet de Minix

What would you like to see most in minix?

Linus Benedict Torval 25 août 1991 23:12
Send to : comp.os.minix

Hello everybody out there using minix -

I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things).

I've currently ported bash(1.08) and gcc(1.40), and things seem to work. This implies that I'll get something practical within a few months, and I'd like to know what features most people would want. Any suggestions are welcome, but I won't promise I'll implement them :-)

Linus (torv...@kruuna.helsinki.fi)

PS. Yes - it's free of any minix code, and it has a multi-threaded fs. It is NOT protable (uses 386 task switching etc), and it probably never will support anything other than AT-harddisks, as that's all I have :-(.

Un autre petit message sur le forum

LINUX is obsolete

ast 30/01/92 09:04 ast 30 janvier 1992 09:04
Send to : comp.os.minix

....
....

I still maintain the point that designing a monolithic kernel in 1991 is
a fundamental error. Be thankful you are not my student. You would not
get a high grade for such a design :-)

....
....

Prof. Andrew S. Tanenbaum (a...@cs.vu.nl)

Petite chronologie des versions du noyau Linux

1994 **1.0** : un noyau stable qui assure les services d'un UNIX classique

1995 **1.2** : beaucoup plus d'architectures processeur, modules chargeables

1996 **2.0** : PowerPC, Multiprocesseur, amélioration de la gestion du réseau, Tux

1999 **2.2** : Framebuffer, NTFS, Joliet, IPv6

2001 **2.4** : mécanisme de writeback, USB, PCMCIA, I2O, NFS 3, X86-64

2003 **2.6** : **changement majeur le noyau devient préemptible** ALSA, ACL, NFS 4

2006 **2.6.18** : lockdep, priority inheritance, ordonnanceur d'E/S CFQ

2007 **2.6.20** : virtualisation KVM

2007 **2.6.22** : allocateur de mémoire SLUB

2008 **2.6.26** : intégration du débogueur du noyau kgdb

2009 **2.6.29** : intégration de Btrfs, KMS (Kernel-based mode-setting)

2011 **3.0** : **il n'y a pas changement majeur** nouvelle mise en cache des pages mémoire

2012 **3.2** : writeback dynamique, contrôle de bande passante dans l'ordonnanceur

2012 **3.3** : "naturalization" du contrôleur mémoire par groupe (memcg)

2013 **3.8** : optimisation pour architecture NUMA

2013 **3.11** : compression des pages de swap

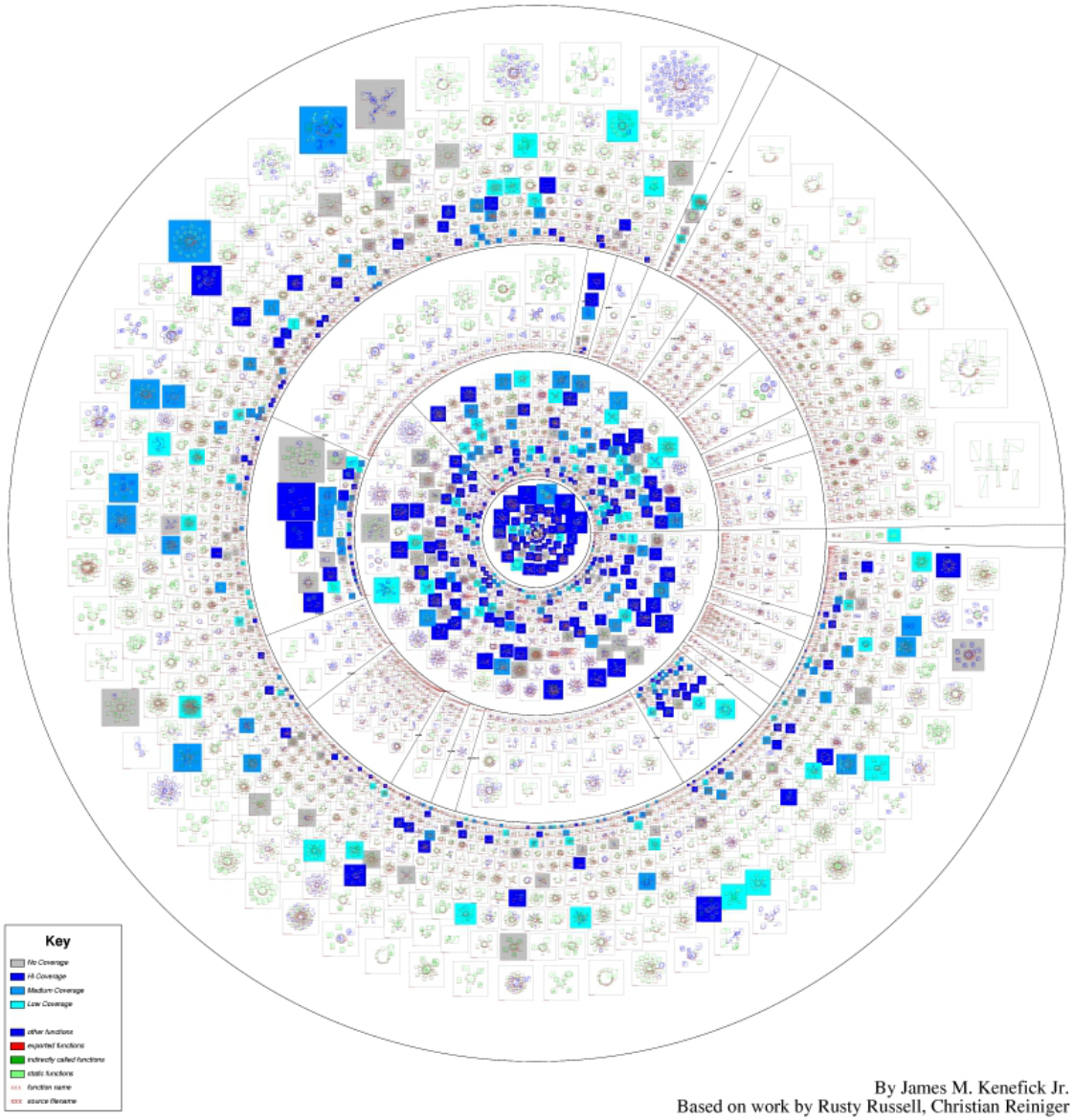
2014 **3.13** : équilibrage NUMA automatique, nftables

2014 **3.17** : file-sealing (vers la fin des toilettes norvégienne ...)

Outline

Taxinomie des noyaux
Histoire de Linux
Architecture du noyau Linux
Structuration des sources du noyau

Linux une Architecture en oignon



Les fonctionnalités du noyau Linux

Un noyau Linux offre 5 grandes fonctionnalités :

1. Gestion des processus
2. Gestion de la mémoire
3. Gestion des services réseaux
4. Gestion du stockage
5. Gestion de l'interface avec l'utilisateur

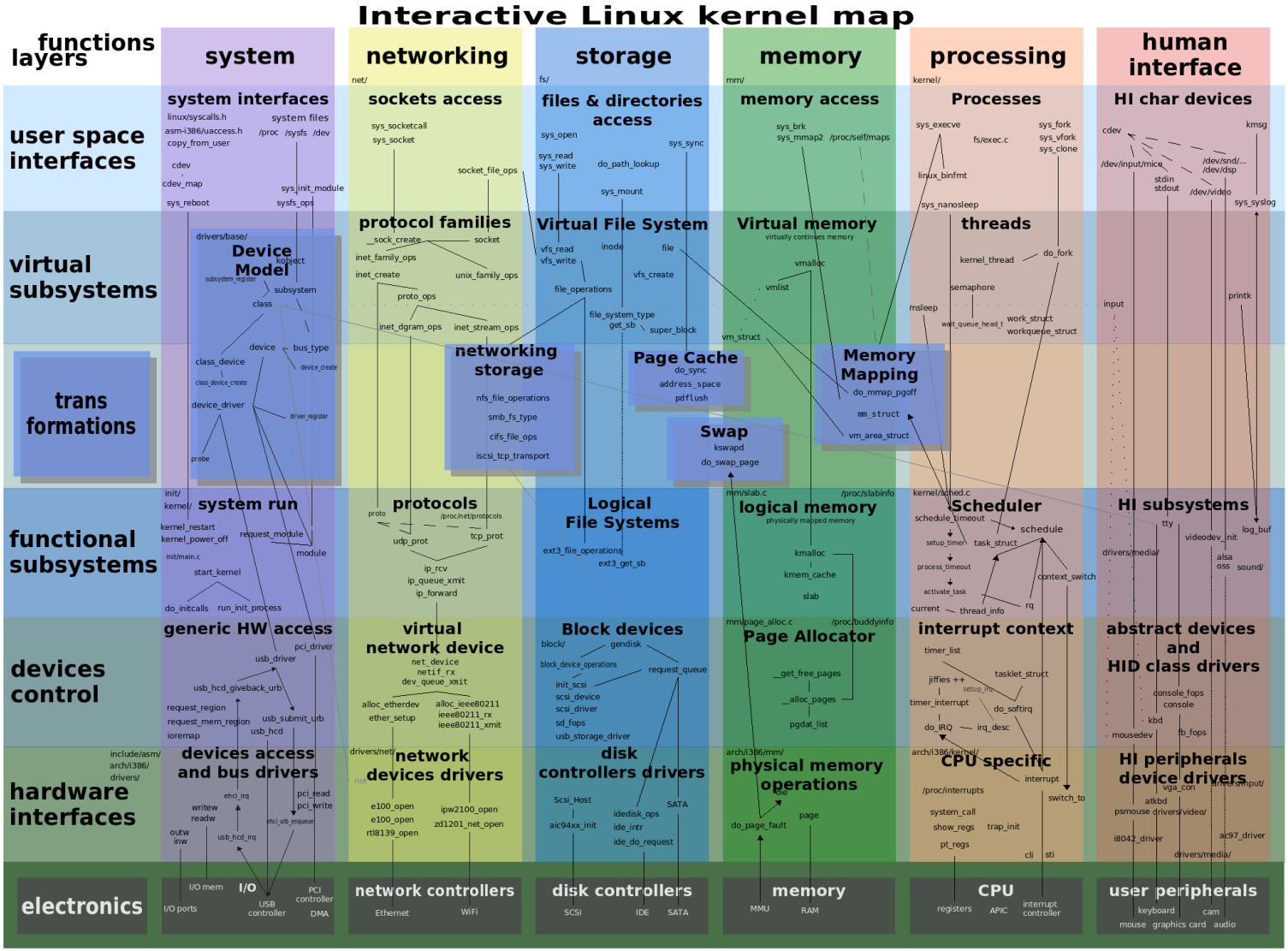
Les couches du noyau Linux

Pour chacune des fonctionnalités du noyau Linux, on retrouve la même série d'abstractions :

1. Couche d'interaction avec le matériel
2. Couche d'abstraction du matériel
3. Couche d'interopérabilité entre les fonctionnalités
4. Couche de services
5. Couche d'abstraction des services

Vue simplifiée de l'architecture du noyau Linux

Diagramme matriciel du noyau GNU Linux



Outline

Taxinomie des noyaux
Histoire de Linux
Architecture du noyau Linux
Structuration des sources du noyau

Structuration Générale

Les sources du noyau sont réparties dans plusieurs répertoires :

```
arch/
block/
COPYING
CREDITS
crypto/
Documentation/
drivers/
firmware/
fs/
include/
init/
ipc/
Kbuild
Kconfig
kernel/
lib/
MAINTAINERS
Makefile
mm/
net/
README
REPORTING-BUGS
samples/
scripts/
security/
sound/
tools/
usr/
virt/
```

Structuration Générale

Les sources du noyau sont réparties dans plusieurs répertoires :

```
arch/
block/
COPYING
CREDITS
crypto/
Documentation/
drivers/
firmware/
fs/
include/
init/
ipc/
Kbuild
Kconfig
kernel/
lib/
MAINTAINERS
Makefile
mm/
net/
README
REPORTING-BUGS
samples/
scripts/
security/
sound/
tools/
usr/
virt/
```

On peut les regrouper en quatre grandes familles :

1. Utilitaires et environnement du noyau
2. Cœur des sources du noyau
3. Fonctionnalités spécifiques
4. Drivers et codes spécifiques aux architectures

Structuration Générale

Les sources du noyau sont réparties dans plusieurs répertoires :

```
                                    samples/
                                    scripts/

Documentation/

                                    tools/
                                    usr/
```

On peut les regrouper en quatre grandes familles :

1. Utilitaires et environnement du noyau

Structuration Générale

Les sources du noyau sont réparties dans plusieurs répertoires :

```
                    init/
                                          samples/
                                          scripts/
                    kernel/
Documentation/      lib/
                                          tools/
                                          usr/

include/
```

On peut les regrouper en quatre grandes familles :

1. Utilitaires et environnement du noyau
2. Cœur des sources du noyau

Structuration Générale

Les sources du noyau sont réparties dans plusieurs répertoires :

```
                    init/
block/              ipc/
                                        samples/
                                        scripts/
crypto/             kernel/             security/
Documentation/      lib/                sound/
drivers/                                tools/
                                        usr/
fs/                 mm/                 virt/
include/            net/
```

On peut les regrouper en quatre grandes familles :

1. Utilitaires et environnement du noyau
2. Cœur des sources du noyau
3. Fonctionnalités spécifiques

Structuration Générale

Les sources du noyau sont réparties dans plusieurs répertoires :

```
arch/                   init/
block/                  ipc/
                                        samples/
                                        scripts/
crypto/                 kernel/         security/
Documentation/          lib/            sound/
drivers/                                tools/
firmware/                               usr/
fs/                     mm/             virt/
include/                net/
```

On peut les regrouper en quatre grandes familles :

1. Utilitaires et environnement du noyau
2. Cœur des sources du noyau
3. Fonctionnalités spécifiques
4. Drivers et codes spécifiques aux architectures

Structuration Générale

Les sources du noyau sont réparties dans plusieurs répertoires :

```
arch/
block/
COPYING
CREDITS
crypto/
Documentation/
drivers/
firmware/
fs/
include/
init/
ipc/
Kbuild
Kconfig
kernel/
lib/
MAINTAINERS
Makefile
mm/
net/
README
REPORTING-BUGS
samples/
scripts/
security/
sound/
tools/
usr/
virt/
```

On peut les regrouper en quatre grandes familles :

1. Utilitaires et environnement du noyau
2. Cœur des sources du noyau
3. Fonctionnalités spécifiques
4. Drivers et codes spécifiques aux architectures

Structuration des sources du noyau (1)

Utilitaires et environnement du noyau :

Documentation : cette documentation au format text s'ajoute aux commentaires du code

scripts : scripts utilisés lors de la configuration du noyau

usr : codes de l'utilitaire `gen_init_cpio` servant à générer l'image vmlinuz

tools : utilitaire permettant d'interagir avec le noyau

samples : ensembles d'exemples simples, mais aussi des modules en déshérence (parfait points de départ pour commencer à contribuer).

Cœur des sources du noyau :

init : gestion du démarrage du noyau, dont le fichier `main.c` qui est le code principal et relie tous les autres fichiers

kernel : codes génériques principaux du noyau

lib : libC utilisée pour compiler le noyau

include : les headers nécessaires à la compilation du noyau et des modules

Structuration des sources du noyau (2)

Fonctionnalités spécifiques :

mm : codes de gestion de la mémoire

block : codes pour les drivers des block-devices, i.e. les devices qui permettent un accès aux données par morceaux et non par flux continus

fs : codes des systèmes de fichiers supportés

ipc : codes pour la communication inter-processus noyau

net : codes pour le support réseau

security : codes des mécanismes de sécurité du noyau

sound : drivers des cartes son/audio

crypto : implémentation de nombreux algorithme de cryptographie

virt : codes permettant de faire de linux un hyperviseur. Actuellement celui de `kvm`. Ceux pour le fonctionnement au-dessus d'un hyperviseur sont dans `drivers` (par exemple `xen`)

Structuration des sources du noyau (3)

Drivers :

drivers : codes pour différents éléments matériels

firmware : codes permettant d'interpréter les signaux provenant des devices

Codes spécifiques aux architectures :

arch : codes spécifiques à chaque architecture supportée