

Introduction aux systèmes GNU/Linux

Séance 1

inetdoc.net



Philippe Latu / Université Toulouse 3 - Paul Sabatier
Document sous licence GNU FDL v1.3
<http://www.gnu.org/licenses/fdl.html>

Plan & Objectifs

- Enseignements systèmes GNU/Linux

- Progression en 3 modules

- 1 - Administration d'un système individuel

- 2 - Administration système en réseau

- 3 - Système d'interconnexion réseau

- Objectifs du module 1 : Introduction aux systèmes GNU/Linux

- Définir

- Noyau Linux

- Logiciel Libre

- Distribution

- Installer et utiliser un système GNU/Linux

- Gérer les paquets de la distribution Debian

- Gérer les comptes utilisateurs locaux et les droits associés

Plan Introduction aux systèmes GNU/Linux

- Séance 1 – Du système Unix aux distributions GNU/Linux
 - Présentation progression
 - Concepts Unix, GNU/Linux, projets Open-Source
 - Paquets applicatifs & Distributions
 - Méthode de travail
- Séance 2 - Installation du Système GNU/Linux
 - Partitionnement et formatage d'un disque dur
 - Installation du système de base
 - choix des paquets d'applications
 - Installation du gestionnaire d'amorce

Plan Introduction aux systèmes GNU/Linux

- Séance 3 – Interfaces graphiques et gestion de paquets
 - Environnements graphiques et chaînes de développement
 - Outils de gestion de paquets
 - Identification des composants d'un paquet
 - Application à la distribution Debian
- Séance 4 – Shell, processus et compilation d'une application
 - Présentation du shell Bash et du langage de script
 - Gestion de processus
 - Gestion des droits sur le système de fichiers
 - Compilation d'une application à partir de ses sources

Plan Introduction aux systèmes GNU/Linux

- Séance 5 – Comptes utilisateurs – uids/gids – syslog & cron
 - Gestion et personnalisation des comptes utilisateurs
 - Rôle des identifiants d'utilisateur et de groupe
 - Journalisation système avec syslog
 - Planification des tâches avec cron
- Séance 6 – Initialisation d'un système d'exploitation
 - Présentation des étapes avant amorçage
 - Rôle du gestionnaire d'amorce
 - Introduction aux niveaux de démarrage
 - Définition des espaces mémoire noyau et utilisateur

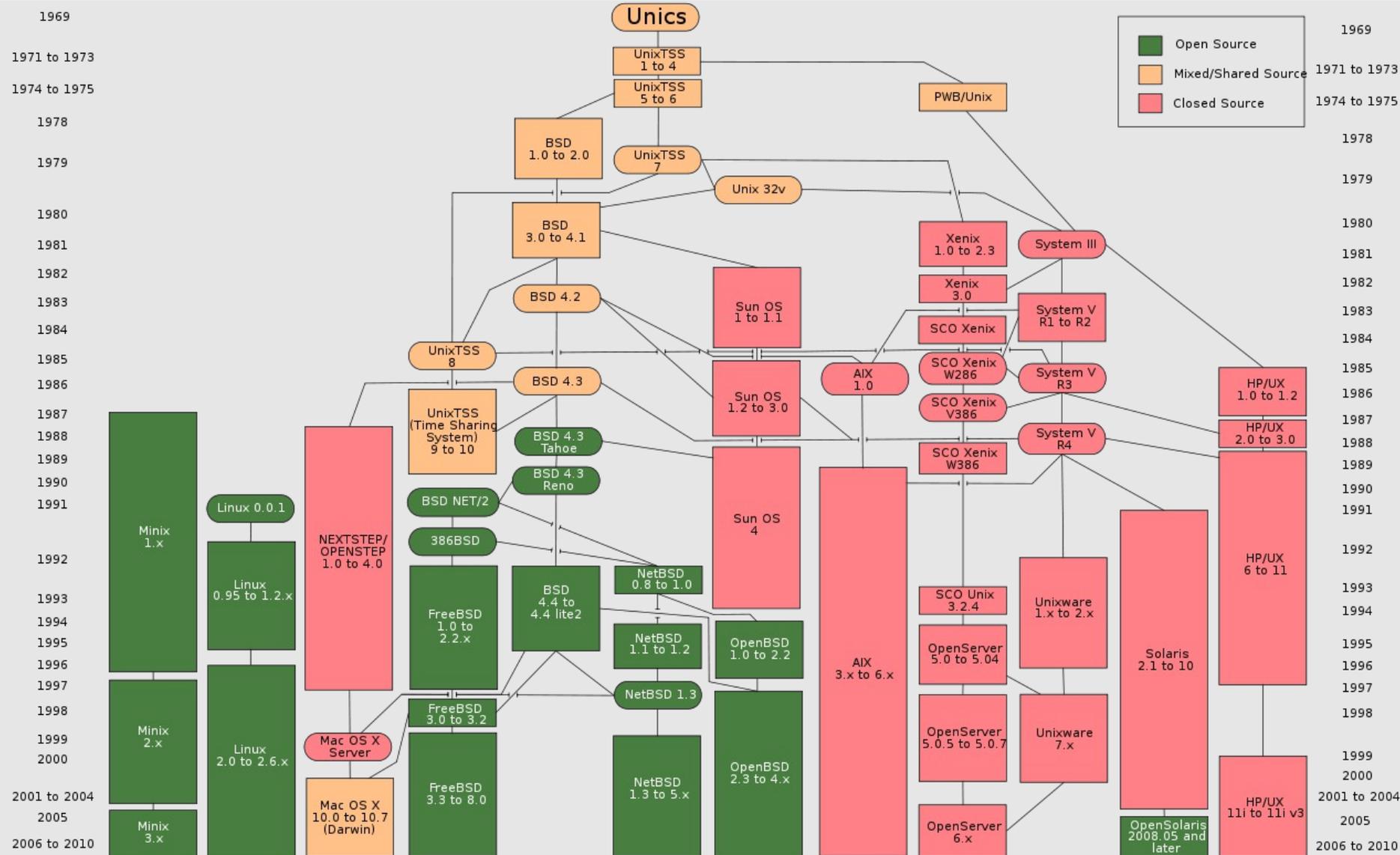
Concepts Unix et GNU/Linux

- Pourquoi étudier le logiciel libre ?
 - Histoire cohérente et continue sur 3 décennies
 - Outil d'analyse critique
 - Processus d'assurance qualité original
 - Processus métier original
- Objectifs
 - Identifier les fonctions de base des système
 - Connaître les grandes étapes de l'histoire des systèmes Unix
 - Identifier les fonctions spécifiques aux systèmes GNU/Linux
 - Identifier les différences entre les principales licences

Concepts Unix et GNU/Linux

- 5 fonctions de base des systèmes Unix
 - **Multi-tâches**
 - Temps processeur partagé entre plusieurs programmes
 - **Multi-utilisateurs**
 - Système partagé entre plusieurs utilisateurs
 - **Portabilité**
 - Outils système partagés entre ordinateurs différents
 - **Bibliothèques de développement standard**
 - Optimisation des développements en partageant du code source
 - **Applications communes**
 - Services système, services Internet, etc.

Concepts Unix et GNU/Linux



Source wikipedia
<http://en.wikipedia.org/wiki/Unix>

Concepts Unix et GNU/Linux

- 1969 – Unics – AT&T – Système V
 - **Unix est un système «accidentel»**
 - AT&T Bell labs – Ken Thompson - Dennis Ritchie
 - 1973 réécriture en Langage C
 - Diffusion sous licence AT&T incluant la totalité du **code source**
 - 1975 publication RFC681 **NETWORK UNIX**
 - **Apparition des variantes propriétaires**
 - Coût de licence prohibitif
 - Versions constructeurs incompatibles entre elles
 - Segmentation en parts de marché captives



Concepts Unix et GNU/Linux

- 1977 – Berkeley University – BSD
 - **Branche Unix lancée à partir d'une licence AT&T**
 - Nombreuses améliorations
 - Gestion mémoire
 - Sous-système réseau TCP/IP
 - Diffusion entre universités
 - Développement de l'Internet universitaire
 - **Procès AT&T**
 - BSD est devenu un système complet autonome
 - Éclatement de la branche BSD
 - FreeBSD, NetBSD et OpenBSD

Concepts Unix et GNU/Linux

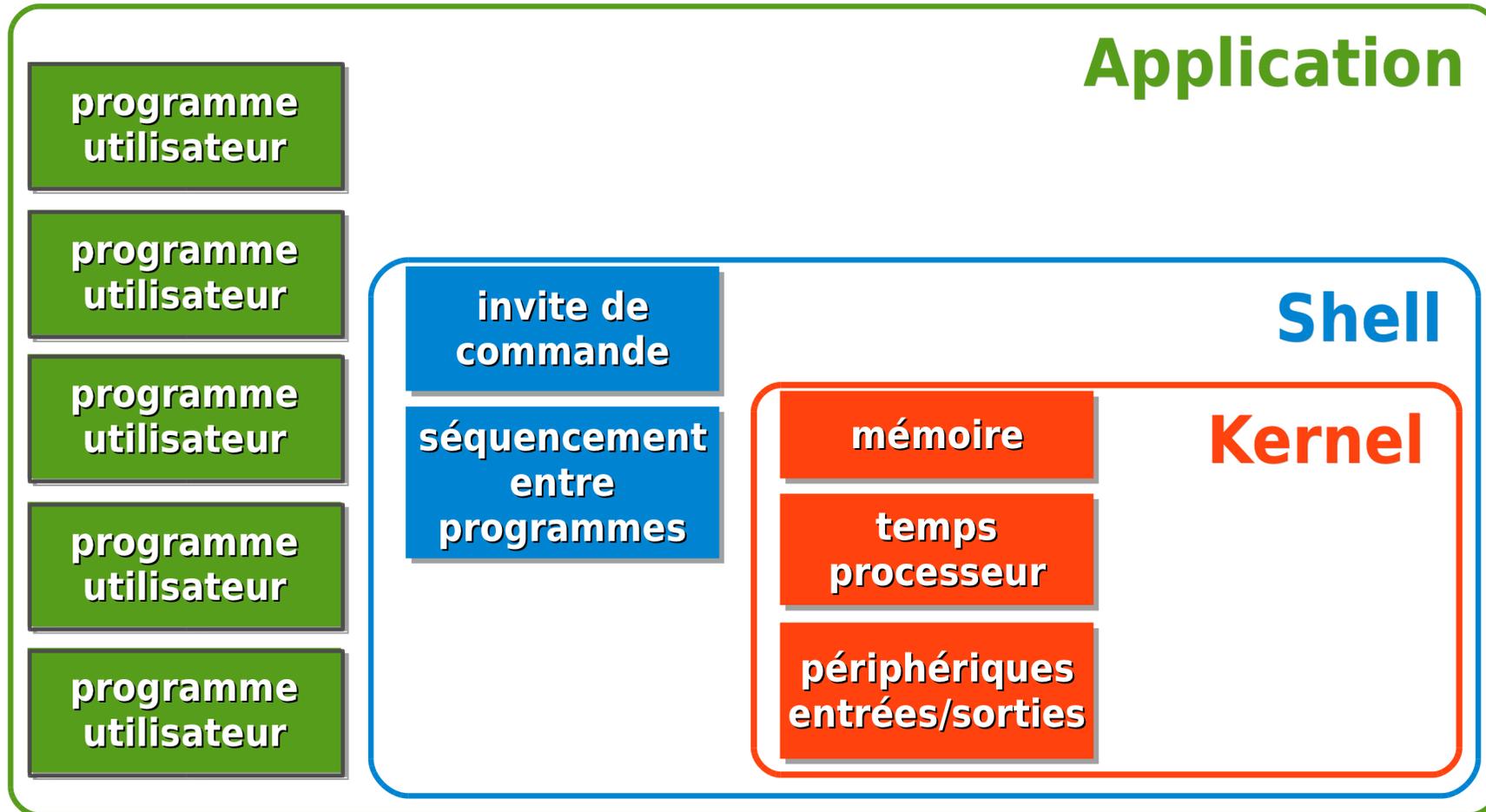
- 1984 – GNU – Not Unix
 - **Projet lancé par Richard Stallman**
 - **2 objectifs**
 - Promouvoir le développement des logiciels libres
 - Protection des travaux des développeurs à l'aide de licences spécifiques
 - Fédérer les développements libres
 - Applications GNU
 - **Unix comme modèle**
 - Fonctions de base déjà éprouvées
 - 1990 – chaîne de développement stable
 - GNU Compiler Collection
 - 1991 – arrivée du noyau Linux ... la pièce qui manquait à l'édifice

Concepts Unix et GNU/Linux

- 1991 – Début du noyau Linux – (noyau = kernel)
 - **Développement initié par Linus Torvalds**
 - **«divergences de vues» avec A.S. Tanenbaum**
 - correctifs sur le noyau Minix refusés
 - <http://www.oreilly.com/catalog/opensources/book/appa.html>
 - **Fonctions de base Unix + quelques spécificités**
 - Multi-tâches
 - Multi-utilisateurs
 - Gestionnaire mémoire
 - Mémoire virtuelle = utilisation répétitive et étendue
 - Mode protégé (processeurs Intel)
 - Contrôle d'accès

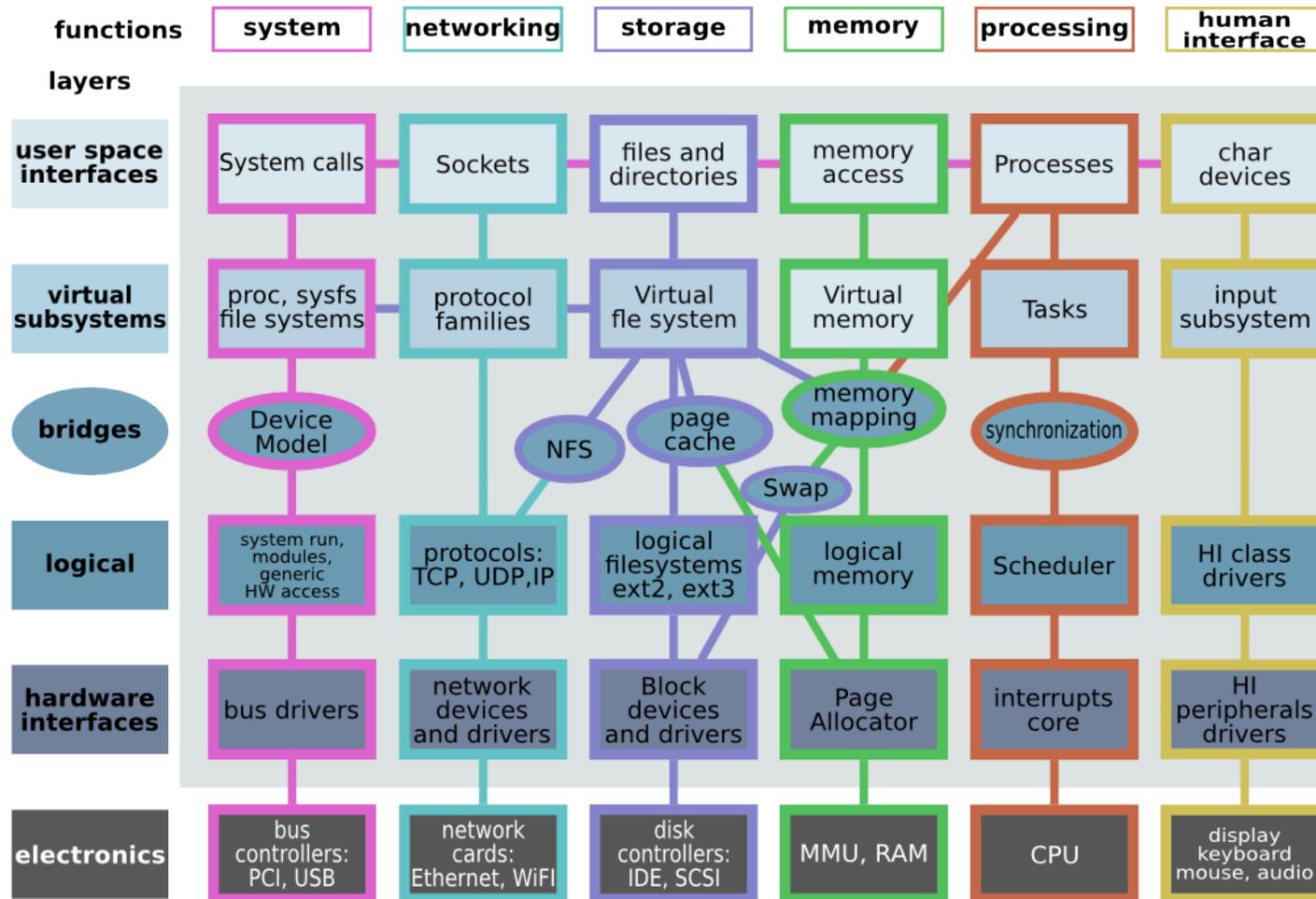
Concepts Unix et GNU/Linux

▪ Système GNU/Linux



Noyau Linux

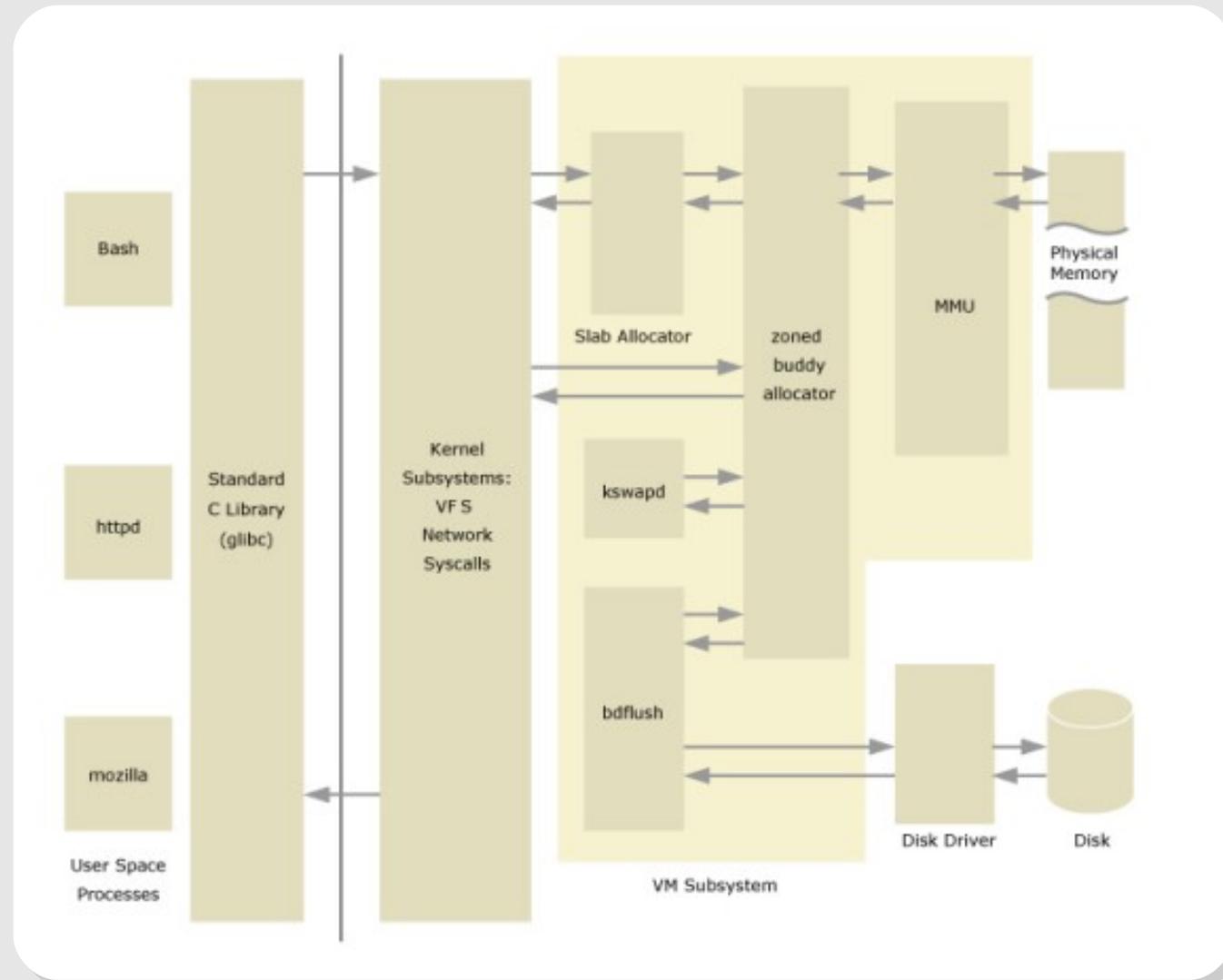
Linux kernel diagram



© 2007-2009 Constantine Shulyupin <http://www.MakeLinux.net/kernel/diagram>

Noyau Linux

- Mémoire virtuelle
 - **userspace**
 - Programmes utilisateurs
 - Bibliothèques standard glibc
 - **kernelspace**
 - Memory Management Unit (MMU)
 - Zoned buddy allocator
 - Allocation pages mémoires
 - Slab allocator
 - Gestion de cache dans les pages mémoire
 - Kernel threads
 - Réutilisation de la mémoire allouée



Noyau Linux

- Ordonnanceur - Scheduler

- **3 domaines ou types de tâches**

- **Domaine temps réel**

- Contraintes de temps d'exécution élevées
 - Fréquence d'exécution garantie

- **Domaine entrées/sorties**

- Attente de disponibilité des périphériques

- **Domaine CPU**

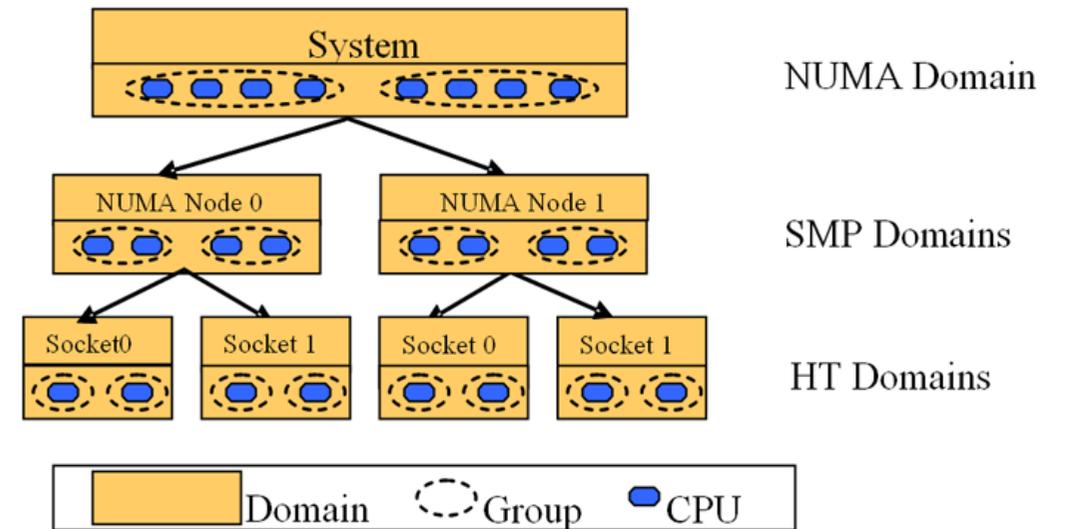
- Temps consacré aux calculs

- **Tranche de temps CPU - time slice**

- Durée d'exécution d'un processus sur un cœur

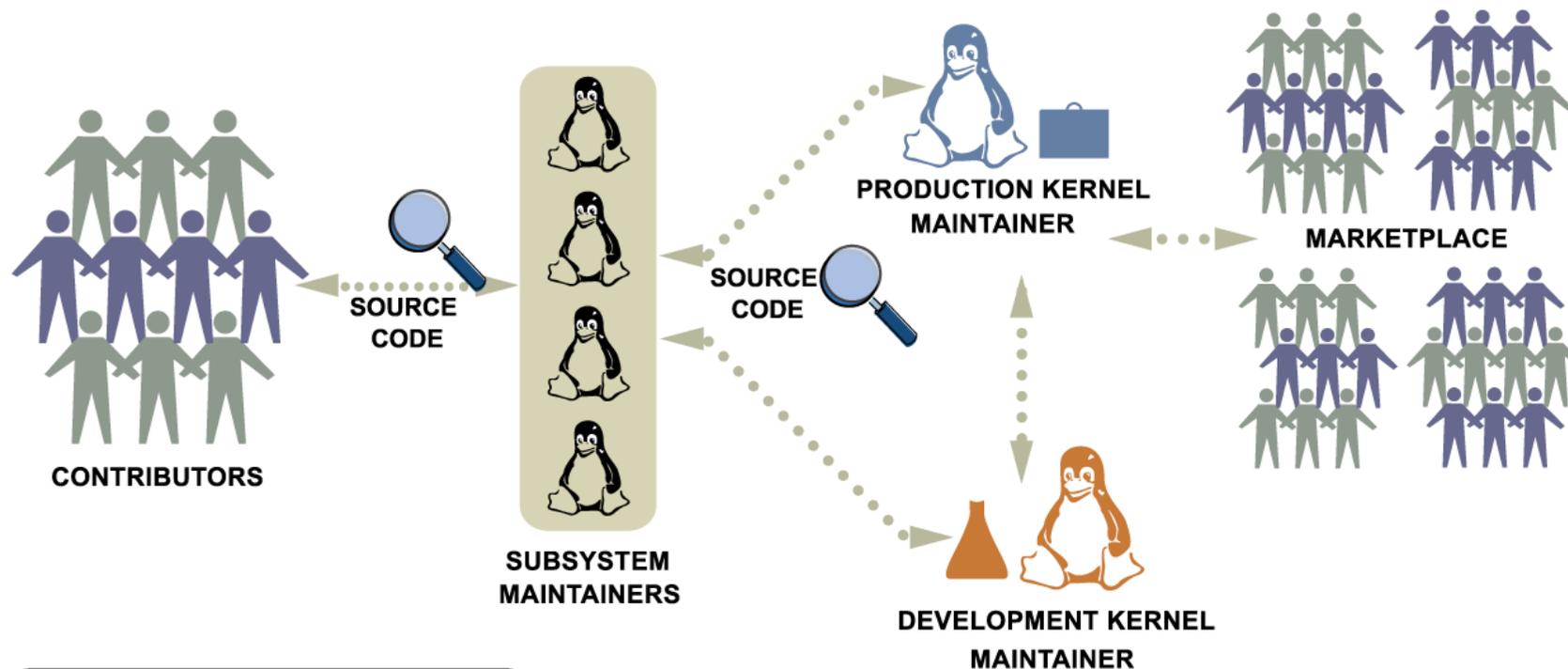
- **Préemption**

- Interruption d'un processus par un second de priorité plus élevée



Noyau Linux

LINUX KERNEL DEVELOPMENT PROCESS

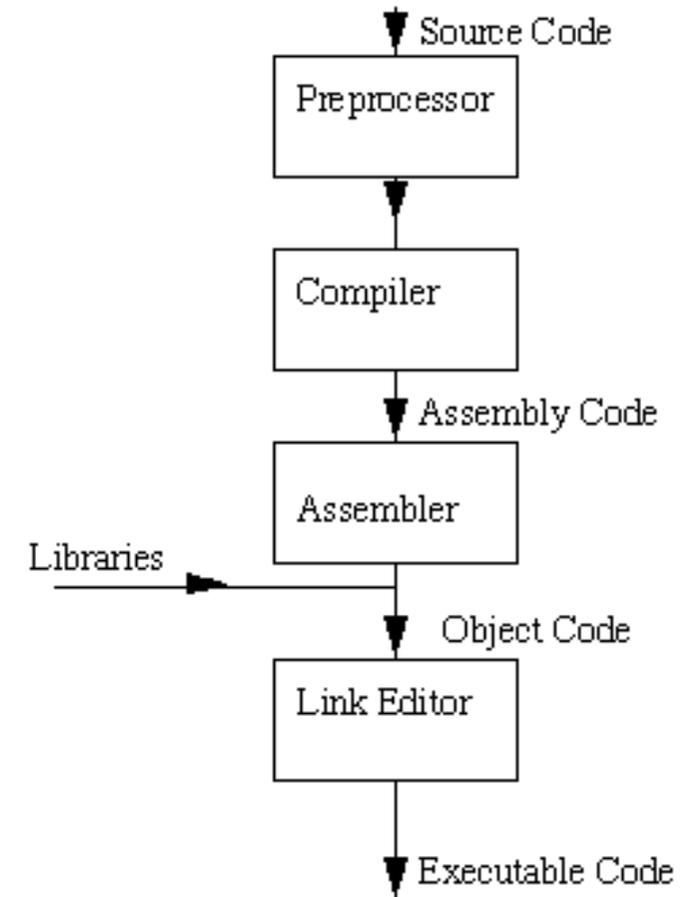


Ongoing peer review of code
Continuously available online
for public review

© 2003 Open Source Development Labs

Logiciel Libre & Licences

- Code source → code exécutable
 - **Tout programme est écrit dans un langage**
 - Exemple : le noyau Linux est écrit en Langage C
 - Le code source n'est pas directement utilisable
 - **Compilation**
 - Transformation du code source en code exécutable
 - Transformation inverse «impossible»
 - Code exécutable = binaire
 - **Logiciel propriétaire**
 - Droit d'utilisation limité d'un code exécutable
 - **Logiciel libre**
 - Accès au code source
 - Droit d'utilisation, d'échange, de modification et de redistribution



Logiciel Libre & Licences

- Licences de Logiciel Libre
 - **Licence BSD → restrictions possibles**
 - Création de versions propriétaires autorisée
 - Restrictions possibles sur les droits de redistribution
 - Restrictions rarement appliquées dans les faits
 - **Licence GNU → Copyleft**
 - Copyright != Copyleft
 - Principe de protection du Logiciel libre et des concepteurs
 - Restrictions interdites sur les conditions de redistribution



- Application du Copyleft
 - **Appliquer un copyright sur le logiciel**
 - **Fixer les conditions de distribution**
 - «Donner à tout utilisateur le droit d'utiliser, de modifier et de redistribuer le programme sans changer les conditions de distribution
 - **Le code source et les libertés associées sont inséparables**
 - <http://www.gnu.org/copyleft/copyleft.fr.html>



Projets OpenSource

- Applications OpenSource & systèmes GNU/Linux

- **Développements parallèles**

- Évolutions, processus et méthodes

- **1er exemple : les services Internet**

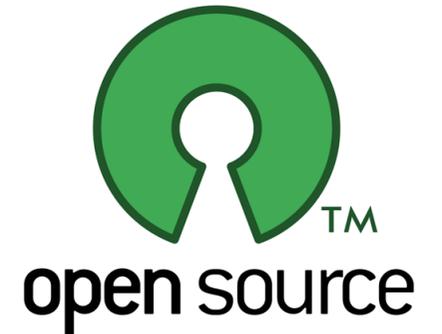
- Bind : noms de domaines
 - Postfix : courrier électronique
 - Apache : serveur web
 - <http://survey.netcraft.com>

- **Terminologie OpenSource**

- Plus que la simple diffusion du code source Libre redistribution
 - Modifications distribuées dans les mêmes conditions que l'original
 - Restrictions possibles sur la redistribution des correctifs
 - Licence sans restrictions sur d'autres logiciels associés



POSTFIX



Projets OpenSource

- Modèle de développement
 - **Conditions uniques !**
 - Capitalisation de compétences sur la durée (~40 ans)
 - Processus d'assurance qualité éprouvé
 - Population de développeurs très importante
 - **Écosystèmes biens structurés**
 - Grands acteurs : Google, Intel, IBM
 - Fondations : Mozilla, LibreOffice
 - <http://www.opensource.org/>



Projets OpenSource

- Communautés & Outils de travail collaboratif

- **Développement**

- Services en ligne - github et gitorious
 - Fermes de compilation
 - <http://savannah.gnu.org/>
 - <http://alioth.debian.org/>

- **Répertoires de projets OpenSource**

- Métriques : qualité, popularité, dynamique de développement
 - <http://freecode.com/>
 - <http://sourceforge.net/>



Distribution

- Distributions GNU/Linux & BSD
 - **Canaux de diffusion du logiciel libre**
 - **Distribution GNU/Linux = association**
 - Noyau
 - Un ou plusieurs Shells
 - Un ensemble d'applications
 - **Composants distribués sous forme de paquets**
 - Code binaire exécutable
 - Configuration type
 - **Gestion de paquets**
 - Principal enjeu dans la vie d'une distribution
 - <http://distrowatch.com/>

Distribution

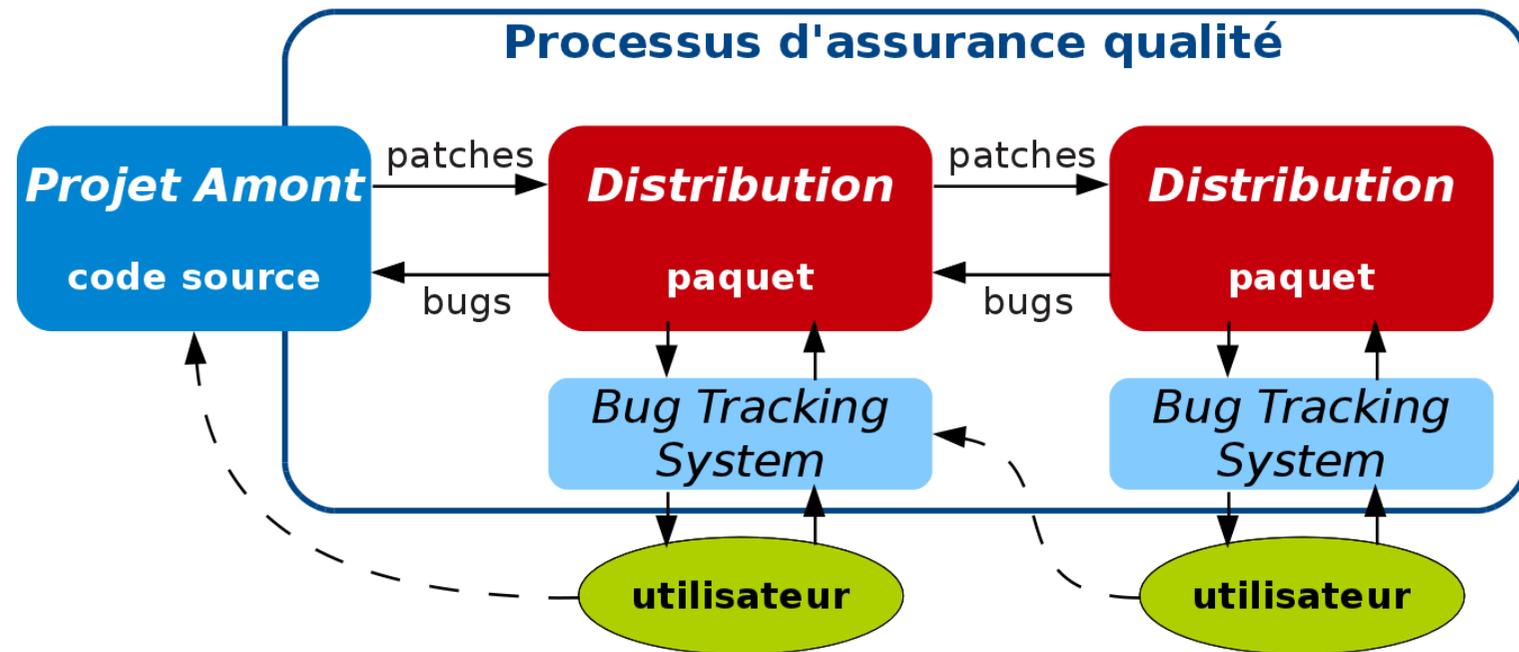
- 2 logiques s'opposent
 - **Publier très régulièrement → utilisateurs**
 - Fournir les outils les plus récents
 - Fournir l'interface la plus attrayante
 - **Publier en fonction de la qualité → infrastructure**
 - Fournir les outils qui satisfont les critères de qualité
 - Garantir la continuité de service
 - **Choix d'une application parmi n**
 - Équivalence entre **qualité du code** et **gestion de paquet**
 - **Responsable de paquet : un rôle essentiel**
 - Capitalisation des compétences d'exploitation
 - Qualités humaines dans la coordination
 - Démarche qualité lors des évolutions

Distribution

- Choix d'une distribution
 - **2 critères essentiels pour l'exploitation**
 - **Facilité d'adaptation**
 - Obtenir une configuration type par contexte
 - Bénéficier de l'expérience des responsables de paquets
 - **Continuité lors des mises à jour**
 - Cohérence des évolutions et corrections
 - Réinstallation impossible
 - **Adaptation + évolution continue**
 - Continuité de service
 - Haute disponibilité

Distribution

- Relations & filiations entre distributions
 - **Assurance qualité & coordination**
 - Exemple : Debian → Ubuntu → LinuxMint
 - Modèle «pipeline»



Distribution

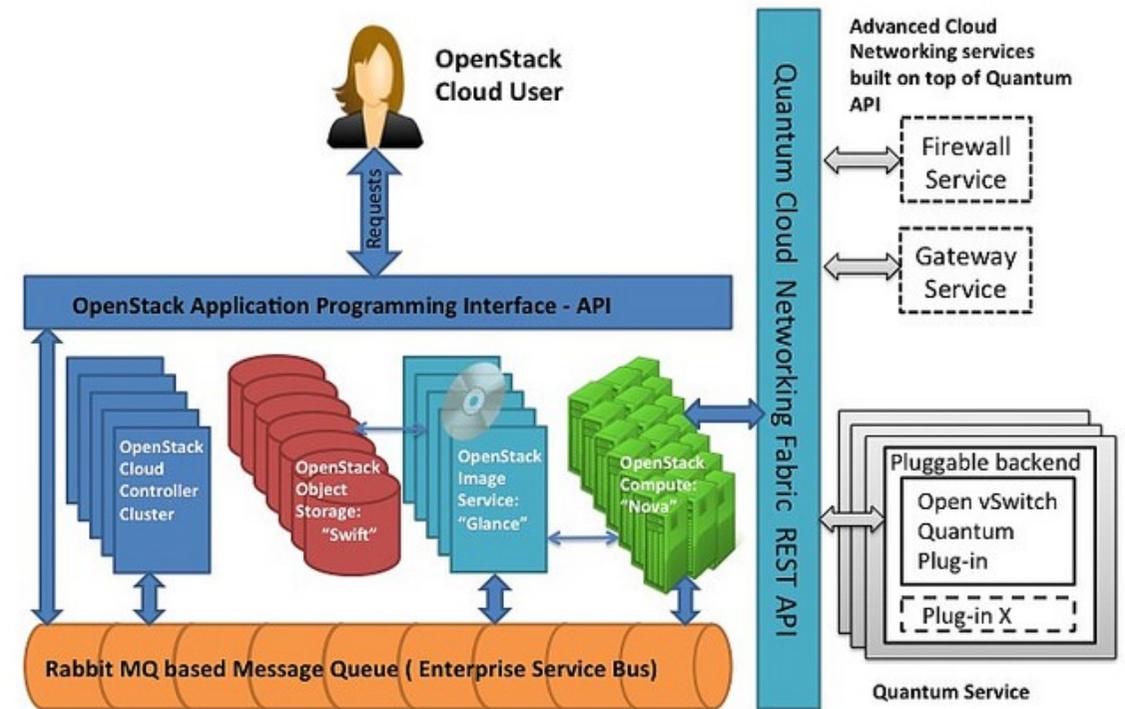
- Debian GNU/Linux
 - **Évolution cohérente et continue depuis 1993**
 - **Contrat social + Principes du logiciel libre selon Debian**
 - Règles à suivre pour garantir qu'un logiciel est bien libre
 - http://www.debian.org/social_contract.html
 - **Charte Debian**
 - Procédure qualité du projet
 - <http://www.debian.org/devel/index.fr.html>
 - **Gestionnaire de paquets APT**
 - Synthèse de toutes les qualités de la distribution
 - Continuité indépendante des versions
 - Adaptabilité en séparant la configuration de l'application
 - Automatisation de la publication des correctifs de sécurité

Distribution

- Debian GNU/Linux
 - **Choix pédagogique**
 - Mille et unes distributions spécialisées
 - Très peu de distributions généralistes
 - **Transparence des processus**
 - Qualité : <http://packages.qa.debian.org>
 - Sécurité : <http://www.debian.org/security/>
 - **Processus métiers difficiles à illustrer**
 - Coût d'accès au support
 - Diffusion restreinte de l'information
 - Difficulté d'accès à l'information
 - Communautés peu ouvertes

Bilan séance 1

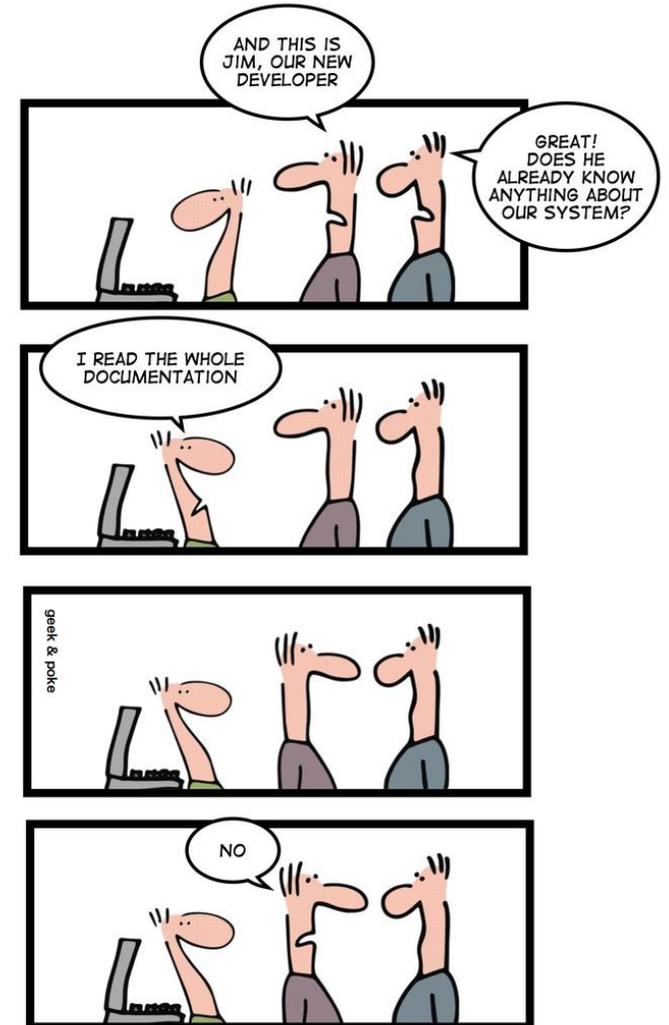
- Logiciels Libres & Projets OpenSource
 - **Rôle majeur dans le monde des technologies de l'information**
 - Modèle de développement unique et éprouvé
 - «La Cathédrale» vs. «Le Bazar»
 - **Écosystèmes innovants**
 - Exemple : OpenStack
 - <http://openstack.org/community/companies/>
 - **Espaces d'échange**
 - **Espaces d'accès à l'information**
 - **L'Internet a toujours progressé avec les logiciels libres**



OpenStack + Quantum Integration Architecture

Bilan séance 1

- Étapes du développement des systèmes Unix
 - **Histoire «continue» sur plus de 40 ans**
 - Histoire + Mémoire = culture
 - Opposition entre culture et obscurantisme
 - Savoir-faire != «recettes de cuisine»
 - **Coût d'apprentissage important**
 - Investissement sur le long terme
 - Capitalisation des savoir-faire = autonomie
 - **Il faut être motivé !**



Ressources

- Cahier de l'Admin Debian
 - <http://raphaelhertzog.fr/livre/cahier-admin-debian/>
- Formation Debian GNU/Linux
 - <http://formation-debian.via.ecp.fr/>
- Framabook
 - Unix. Pour aller plus loin avec la ligne de commande
 - <http://framabook.org/unix-pour-aller-plus-loin-avec-la-ligne-de-commande>
- Manuel d'installation Debian
 - <http://www.debian.org/releases/stable/installmanual>

