

# Le projet Linux est-il un modèle possible d'entreprise innovante ?

*Author : jul*

*Id : report.tex, v2.12002/04/0212 : 45 : 38julExp*

© Julien Tayon 2002

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 ; with the Invariant Sections being everything for now, A copy of the license is available at <http://www.gnu.org/licenses/fdl.txt>.

Julien Tayon

2 avril 2002

## Résumé

Ce texte est issu de la correction du document présenté à la chaire “Développement des Systèmes d’Organisation”[1] du CNAM dans le cadre de mon mémoire de probatoire. Il s’agit d’un travail de 5 semaines qui est donc par définition succinct.

*Un des thèmes de Paul Ricoeur est la proposition : l’utopie questionne, là où l’idéologie récupère.*

C’est le plan choisi pour le présent document. Sur un plan purement formel, le projet Linux est considéré ici comme une utopie, et les théories économiques libérales comme une idéologie.

Du fait de sa production, la communauté Linux a prouvé sa capacité à

- produire un noyau selon des normes strictes (Unix),
- réagir rapidement par rapport aux besoins des clients,
- innover.

La structure de cette organisation est considérée comme un *bazar*. Elle semble questionner les théories organisationnelles et économiques sur leurs principes.

La structure du mémoire est la suivante : dans un premier temps les notions de transactions et de contrats informels sont étudiées. Elles sont ensuite comparées aux pratiques du logiciel libre. Elles permettent enfin de mettre en évidence l’originalité de Linux comme structure innovante basée sur les contrats informels (la culture comme mode de gouvernance). La conclusion porte sur les applications possibles, c’est-à-dire les méthodes applicables à l’entreprise, et des règles permettant des relations entre ces deux types d’organisation.

Le document est disponible sur les sites :

- <http://www.libroscope.org/doc/linuxorga/>,
- <http://www.cnam.fr/deps/te/dso/>,
- les transparents sur <http://www.libroscope.org/doc/linuxorga/slides>,
- les sources sont disponibles sur <http://www.julbox.net/>,

Je tiens à remercier pour leurs contributions :

- Raphaël Rousseau (<http://www.april.org/~r4f/>) *vice-chancellor de la FSF Europe*
- Loïc Dachary (<http://www.dachary.org/loic/>) *membre fondateur de la FSF Europe* ;
- Thierry Pinon (<http://www.libroscope.org>) *membre fondateur de Libroscope* ;
- Guillaume Pernaud ;
- Francis Lacoste (<http://www.contre.com>) *membre fondateur de la FSF Canada* ;
- Jean Baptiste Tayon ;
- Christophe Tayon ;
- Miod Vallat (<http://www.advogato.org/person/Miod/>) *développeur OpenBSD* ;
- Benoît Rouits (<http://brouits.free.fr/>) ;
- Gerard Weisbuch (<http://www.lps.ens.fr/~weisbuch/>) *chercheur au LPS de l'ENS* pour ses conversations ;
- Christian P. Momon (<http://www.ocmland.org>) *membre d'apodéline* ;

Document rédigé en  $\LaTeX$ .

# Table des matières

<b>I</b>	<b>Linux est-il un modèle d'entreprise innovante ?</b>	<b>4</b>
<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	Avant Propos . . . . .	5
1.2	Linux comme organisation . . . . .	5
1.3	Linux comme représentant du logiciel libre . . . . .	6
<b>2</b>	<b>Linux en perspective</b>	<b>7</b>
2.1	Y a t'il un intérêt organisationnel à priori ? . . . . .	7
2.2	Perspective historique des théories des organisations . . . . .	7
2.3	Ce que n'est pas Linux . . . . .	7
<b>3</b>	<b>Les règles structurant le projet Linux</b>	<b>9</b>
3.1	Choix des méthodes . . . . .	9
3.2	Origine culturelle . . . . .	9
3.3	Les grandes influences . . . . .	9
3.4	La licence : une règle du jeu nécessaire . . . . .	10
3.4.1	Propriétés de la licence . . . . .	10
3.4.2	Influence de la licence sur les pratiques de la communauté Linux . . . . .	11
3.4.3	Influence de la licence sur la culture . . . . .	11
3.5	Analyse des jeux pratiqués . . . . .	11
3.5.1	Les asymétries . . . . .	11
3.5.2	Gains et pertes . . . . .	12
3.5.3	Conclusions . . . . .	13
<b>4</b>	<b>Conclusions</b>	<b>14</b>
4.1	Linux et l'innovation : un cercle vertueux . . . . .	14
4.2	Linux et l'entreprise . . . . .	14
4.3	Une aberration organisationnelle ? . . . . .	14
4.4	Des mécanismes d'acculturation sont-ils envisageables entre des structures d'entreprises classiques et la communauté Linux ? . . . . .	15
4.5	Linux et la société : cyber-vigilance et <i>hactivism</i> . . . . .	15
<b>II</b>	<b>Bibliographie</b>	<b>16</b>
<b>III</b>	<b>Annexes</b>	<b>19</b>
<b>A</b>	<b>Linux : qu'est ce ?</b>	<b>20</b>
A.1	Un Unix . . . . .	20
A.2	Le projet de Linus Torvalds . . . . .	20
A.3	Une structure virtuelle . . . . .	20
A.4	Le coeur d'un système hétérogène cohérent . . . . .	20
<b>B</b>	<b>Historique du logiciel libre, et de Linux</b>	<b>21</b>
<b>C</b>	<b>Les règles de la communauté Linux</b>	<b>22</b>
C.1	Lancer le projet . . . . .	22
C.1.1	Genèse . . . . .	22
C.1.2	Publication . . . . .	22
C.1.3	annoncer . . . . .	23

C.2	Maintenance du projet et participation à un projet . . . . .	23
C.2.1	Soumettre des contributions . . . . .	23
C.2.2	Intégrer des contributions . . . . .	23
C.2.3	Documenter . . . . .	23
C.2.4	Discuter . . . . .	23
C.3	Rapport avec les utilisateurs . . . . .	24
C.4	Travail en groupe . . . . .	24
C.5	Autorité . . . . .	24
C.5.1	Récompenses . . . . .	24
C.5.2	Pénalités . . . . .	24
C.6	Les développeurs Linux face à l'argent . . . . .	25
<b>D</b>	<b>Vocabulaire</b>	<b>26</b>

## **Première partie**

# **Linux est-il un modèle d'entreprise innovante ?**

# Chapitre 1

## Introduction

### 1.1 Avant Propos

Par souci de clarté, les termes Linux GNU/Linux, et projet Linux sont sommairement détaillés ci-dessous (ils sont détaillés dans le glossaire et en annexe) :

**Linux** il s'agit d'un noyau conforme aux normes UNIX, il permet l'accès aux ressources systèmes d'un ordinateur. Il est comparable en ceci aux kernel32.exe des machines Microsoft Windows, au MSDOS.SYS des machines sous MS-DOS, et aux noyaux UNIX ;

**GNU/Linux** pour que Linux soit utilisable sur une machine il faut lui adjoindre des logiciels. Cet agrégat logiciel est appelé distribution. Pour le distinguer du noyau proprement dit, il est appelé GNU/Linux. Il est fourni avec un ensemble d'outils du projet GNU ( D page 26) qui lui permet d'être utilisable par un individu ;

**Projet Linux** il s'agit de l'ensemble des personnes contribuant par un mode qui leur est propre à la réalisation de Linux. Il sera question de Linux uniquement dans le document présent, et non pas de Linux dans l'acceptation de logiciel libre. Ce sont deux choses totalement différentes.

### 1.2 Linux comme organisation

*Les «organisations» (entreprises agricoles, tertiaires, marchandes ou non) sont constituées d'éléments divers, dont elles combinent l'utilisation afin de produire des biens ou des services. Parmi ces éléments, on distingue, notamment, les ressources humaines, les capacités financières, les moyens matériels (immobilisation, ...) et immatériels (brevets, expériences, savoir faire, ...) [2].*

Linux est aujourd'hui une référence en terme de système d'exploitation ( A page 20). La communauté Linux a pour unique finalité sa production. Le monde politique et organisationnel vante l'entreprise comme modèle de référence de l'efficacité. Cependant ce cas est un précédent ; le projet Linux est une organisation différente de tout ce qui est connu (syndicat, entreprise, structure associative) qui produit son noyau selon des normes parmi les plus strictes de l'industrie (nommément Unix ( A.1 page 20)). Ce projet est une structure informelle pour laquelle la participation se fait sous la forme de contribution bénévole. Comment expliquer qu'une structure qui ne dispose d'aucun moyen propre <sup>1</sup> réussisse aussi bien à produire que des entreprises aux moyens logistiques et humains bien supérieurs ; des entreprises comme Sun, HP, IBM, Compaq, qui produisent des systèmes d'exploitation concurrents ont finalement tous intégré *GNU/Linux*[3] dans leur offre, en renonçant parfois même à leur produit maison (SGI ( D page 27)), IBM, Compaq, Hewlett Packard).

Ainsi, se posent les questions suivantes :

- cette organisation est-elle une aberration ?
- en tant qu'entreprise, puis-je travailler avec ce type d'organisation ?
- puis-je m'inspirer de leurs méthodes<sup>2</sup> ?
- l'utilisation de pratiques du projet Linux, peut-elle apporter un avantage concurrentiel à une entreprise ?

Pour répondre à ces questions, il faut définir l'originalité du projet Linux par rapport aux autres organisations.

Pour les personnes peu familières avec le monde du logiciel libre, un glossaire, ainsi que les références vers un historique de Linux sont fournis.

---

<sup>1</sup> les infrastructures sont prêtées par des universités ou des entreprises, et les tâches d'administration et de maintenance des systèmes sont accomplies par des professionnels bénévoles.

<sup>2</sup>il est à noter que cette question n'est pas triviale étant donné que Linus Torvalds, leader du projet Linux (qui coordonne les efforts de milliers de personnes) est incapable dans une entreprise traditionnelle d'être chef de projet .

### 1.3 Linux comme représentant du logiciel libre

Linux est un projet parmi tant d'autres du logiciel libre. Le logiciel libre est en fait la pérennisation d'habitudes de programmation prises par la communauté universitaire scientifique depuis les années 1970. Sa conceptualisation (sous le terme de logiciel libre) est en partie issue des travaux de R. Stallman dans la recherche d'une reconnaissance et protection de ces méthodes. Aujourd'hui on peut recenser un minimum de 9000 projets logiciels libres[4].

Le méta-projet le plus cohérent reste GNU ( D page 26). Seulement, le projet logiciel libre officiel pour faire un noyau GNU, «*Hurd*», bien que commencé en 1985 (soit 5 ans avant Linux) n'a rien produit de comparable en terme de stabilité et d'utilisabilité. Le projet Linux est non-conforme car il utilise la licence GPL sans en respecter la politique. Notamment un projet pour être officiellement GNU, doit faire céder les copyrights détenus par ses développeurs à la FSF ( D page 26) afin que celle-ci puisse entreprendre des actions en justice le cas échéant. Linux ne respecte pas ce point, et cette pratique tend à se relâcher avec le temps. Il est aussi à noter que nombre de projets logiciels libres de grande taille étaient faits selon le modèle «*cathédrale*» (emacs,BSD). Linux est un peu l'enfant terrible du logiciel libre, et cependant beaucoup de personnes pensent parler de logiciel libre, lorsqu'ils évoquent le nom du projet Linux.

# Chapitre 2

## Linux en perspective

### 2.1 Y a t'il un intérêt organisationnel à priori ?

Dans *voyage au centre des organisations*, H. Mintzberg implique pour toutes les structures envisagées la notion d'autorité centrale de décision relayée au sein de l'entreprise par la ligne hiérarchique (*middle-management*). Cette description trouve des échos dans le terme de *vision de l'entreprise* apportée par le dirigeant à la structure. Certains projets logiciels libres, dont Linux, privilégient la décentralisation, une décentralisation de la décision et de l'exécution des tâches apparemment totale. Ce modèle dit du «*bazar*» par opposition à la «*cathédrale*» que représente les entreprises est à priori un exemple de désorganisation[5]. Si la désorganisation est une réussite, cela remet totalement en cause la légitimité de la fonction d'organisation dans l'entreprise, et il me semble nécessaire de confronter cette discipline à l'exemple, sinon il peut aussi s'agir d'une mauvaise interprétation intéressante à corriger.

### 2.2 Perspective historique des théories des organisations

Dans la théorie micro-économique, les entreprises sont des unités de production individuelle coordonnées sur le marché par le mécanisme des prix. Les entrepreneurs sont rationnels (rationalité pure) et l'information est parfaite. Dans ce cadre, le principal objectif est la maximisation des profits.

H. Coase[6] considère l'entreprise comme une boîte noire (on ne sait pas de quoi elle est faite). Pour Coase, elle est une forme de coordination des transactions alternative au marché sous la forme de l'administration (bureaucratie). La notion même de transaction est coûteuse et c'est le premier rôle de l'entreprise dans le système libéral, que de permettre la transaction. En effet, le simple fait même d'étudier le marché pour connaître le prix auquel on peut vendre est coûteux. Cette fonction de l'entreprise se substitue en cela à la *La main invisible* d'Adam Smith. En ce sens les ajustements sont aussi liés à l'intériorisation dans l'entreprise, par sa structure, de jeux qui ne sont d'ordre ni économique, ni rationnel.

Entre l'entreprise et le marché il existe une forme hybride : le contrat qui décrit des mécanismes de coordination plus complexes que ceux décrits par le marché. Williamson[7] explique que le choix entre les différents types de coordination est basé sur les spécificités des actifs qui sont le sujet des transactions. Plus un actif est spécialisé plus il est souhaitable de le gérer à l'intérieur de l'entreprise, plus un produit est standard plus le marché est adéquat. Le contrat est ainsi une forme matérialisant les conditions particulières des transactions entre acteurs comme l'entreprise et les clients.

Mais l'entreprise n'est pas seulement de la bureaucratie. Azariadis[8] a démontré qu'à l'intérieur des entreprises il y avait aussi des contrats implicites. Le réseau de ces contrats implicites peut être vu selon Kreps[9] comme une culture d'entreprise. Il semble qu'au vu des 50 dernières années, l'organisation ainsi que les théories ont évolué de façon à prendre de plus en plus en compte la partie informelle des organisations. Linux dans ce cadre est singulier : le contrat implicite, c'est-à-dire la culture de l'organisation, gouverne les relations entre individus. C'est le mode principale de régulation des relations entre individus, comme cela sera montré dans la partie suivante.

L'exo-squelette de l'entreprise est ici absent (bureaucratie, infrastructure, services de relation spécialisés). Linux et l'entreprise appartiennent au domaine de l'organisation, comme la limace et l'escargot sont des gastéropodes. Et même s'il est proche du système universitaire, puisqu'il en est issu, il en a ni la bureaucratie, ni la planification. Le projet Linux s'apparente à un système d'action concret.

### 2.3 Ce que n'est pas Linux

La communauté Linux est une organisation en ce sens qu'elle regroupe des personnes se sentant une appartenance commune par la production d'un noyau appelé Linux ( A page 20). Cependant, Linux n'a pas :

- de système de planification centralisé ou décentralisé ;
- de bureaucratie ;
- de peur de l'opportunisme au sens de Williamson, et donc l'information est symétrique ;
- de logique de maximisation des profits en raison de l'approche particulière des droits de propriété ;
- de logique de subordination des personnes telle qu'introduite par un contrat de travail.

Ainsi, notre partie suivante va essayer de déterminer, à partir d'outils adaptés, ce comment est structuré le projet Linux.

## Chapitre 3

# Les règles structurant le projet Linux

### 3.1 Choix des méthodes

La méthodologie est importante car les outils choisis conditionnent le résultat obtenu. En utilisant des outils de type certification *ISO 9001*, on présuppose d'office que l'objet à étudier est une entreprise. En effet «*Chaque instrument est une théorie matérialisée*» (Gaston Bachelard).

La méthode choisie est celle proposée par Michel Crozier[10]. Elle consiste à évaluer les jeux qui existent au sein de l'organisation pour évaluer les situations qui sont gagnantes. L'analyse des situations considérées comme gagnantes permet de déterminer la finalité. La finalité de l'organisation est la résultante des gains récurrents sur l'ensemble des jeux (règles d'échanges). La motivation désigne les gains escomptables sur un plan personnel. La finalité peut être vue comme les gains escomptables par l'union des individualités.

Le travail portera sur les référentiels communs aux membres du groupe[11]. Pour ce faire, les sources d'informations seront issues de la littérature portant sur le sujet ainsi que des échantillons de discussions disponibles sur ce projet[12].

### 3.2 Origine culturelle

L'analyse de l'environnement repose sur le principe que les gains des auteurs, et leurs alternatives sont dictés par leur apprentissage de règles dans leur milieu culturel d'origine. La culture la plus fortement représentée dans le projet Linux est la culture puritaine et/ou protestante.

La forte représentation des américains, des scandinaves, et des allemands est indiscutable au vu de l'étude menée par le département de psychologie de l'université de Kiel[13]. Cet élément peut être, au vu du travail de Max Weber[14], comme déterminant. En effet, la plupart des développeurs de coeur<sup>1</sup> ont commencé comme Linux sur leur temps personnel. L'élaboration de ce projet est en fait issue de la vocation "*Beruf*" d'une minorité de développeurs. Les développeurs influents (Alan Cox, James Simmons, Linus Torvalds, Donald Becker...) sont avant tout considérés comme des («*hackers*»)( D page 26), avant d'être des employés de leurs compagnies respectives (Red Hat, SuSE, Transmeta, NASA). En fait, ces personnes à tous les stades de leurs carrières universitaires ou professionnelles, ont réussi à avoir du temps détaché pour leur projet ( C.6 page 25).

### 3.3 Les grandes influences

Unix est de par l'ouverture «*accidentelle*» de son code, et de par sa conception simple et propre (KISS ( D page 26)), le système d'exploitation le plus utilisé et le plus étudié en université. Le livre de A. Tannenbaum[15] a servi de source d'inspiration à Linus Torvalds en est un exemple majeur. Ce livre est fourni avec un système d'exploitation semblable à Unix que les gens peuvent étudier : *MINIX*.

L'environnement universitaire est évident : une bonne partie des développeurs de coeur ont initié, ou ont fait leur travail dans le cadre de l'université (Linus Torvalds à Helsinki, James Simmons à Buffalo, Remy Card à Paris VI, ...). Ainsi, une des règles dans le cadre de ce projet est que les personnes ont longtemps participé gratuitement aux conférences, et que le prix de l'entrée des conférences ne devait pas être excessive pour les personnes. Les conférences, comme dans le domaine universitaire, sont non seulement l'occasion de présenter de nouvelles méthodes, mais aussi l'occasion de se rencontrer "pour de vrai" et échanger des idées.

La communauté Linux s'est développée à partir de la communication par email. Les gens sont jugés sur ce qu'ils publient, et non sur ce qu'ils sont, ou paraissent être<sup>2</sup>. Ceci est un point primordial, car les développeurs se recrutent d'une manière

<sup>1</sup>développeurs très actifs dont les contributions sont récurrentes.

<sup>2</sup>Ceci est exprimé par le dicton «*On the Internet no one knows you're a dog*» (Quand vous discutez sur Internet, personne ne remarque que vous êtes un chien).

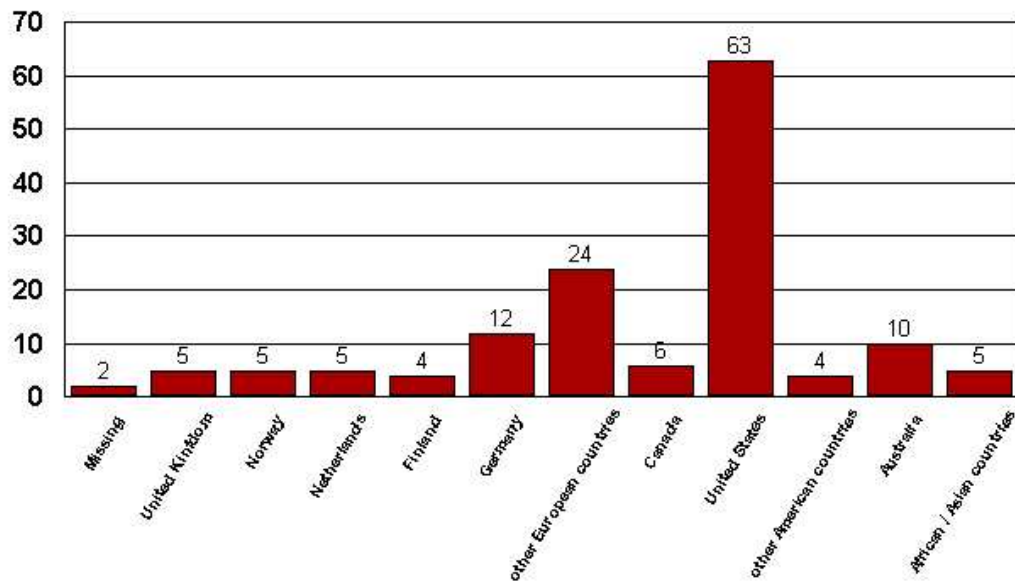


figure 1: Origin of participants

FIG. 3.1 – Répartition par pays des développeurs Linux [13]

importante dans ce que les anglo-saxons appellent la communauté «geek». <sup>3</sup>

L'annonce première de Linus Torvalds présente son projet comme fait pour les «les hommes qui regrettent le temps où l'on devait écrire soi-même ses propres pilotes»[16]. Il se destine initialement aux spécialistes passionnés. La force de Linus Torvalds semble avoir été de jouer dans une idéologie libérale de la volonté de défricher de nouveaux espaces vierges[17] en contrecarrant l'attaque frontale faite par Bill Gates sur ces pratiques dites de "spoliation de la propriété intellectuelle". Cet aspect a notamment été analysé par M. Volle[18].

### 3.4 La licence : une règle du jeu nécessaire

La licence contraint les acteurs de l'organisation dans leurs choix en apportant des éléments pratiques, et idéologiques auxquels ils sont soumis. L'histoire des Unix, et surtout de ceux qui sont libres, ( A.1 page 20) a montré que la licence d'utilisation est un point primordial. Cette partie étudie en quoi la licence structure les jeux pratiqués et quelle influence elle a sur la motivation des développeurs.

#### 3.4.1 Propriétés de la licence

À l'origine, Unix est une création de AT&T( D page 26). À la suite de ses déboires avec la justice pour situation monopolistique dans le domaine téléphonique (violation du "Sherman Antitrust Act"), AT&T fut interdit de développer toute activité commerciale ou non, hors du domaine des télécommunications. Cela autorisait cependant les travaux de recherche, ce qui conduisit à rendre les sources d'UNIX disponibles. L'un de ses auteurs, Ken Thompson, choisit de prendre une année sabbatique pour aller co-développer avec les universitaires son projet dans le cadre de l'université de Berkeley (près de San Francisco). Son travail permit l'élaboration d'un premier clone d'Unix connu comme BSD. Suite au succès de ce projet, AT&T se sentant menacé par l'émergence de ce produit décida de poursuivre ce projet pour violation de copyright[19]. Le rachat du département incriminé par Novell (fabricant de système d'exploitation) mit fin au différend. Novell a soutenu assez tôt les développements de systèmes d'exploitation alternatifs à Microsoft Windows.

Richard Stallman ( D page 26) avait prévu la réaction négative des éditeurs propriétaires aux développements d'un Unix à partir du travail d'un éditeur propriétaire. Il mit en place le développement d'un clone d'Unix en ne réutilisant pas de code source de sociétés propriétaires. Et pour qu'il puisse fonctionner, il fallait que la plupart des outils nécessaires à son fonctionnement et à son existence soient créés (compilateur, shell, bibliothèques de programmation). Le choix d'Unix est lié à la simplicité de sa conception, et à la robustesse que cela lui confère.

La licence GPL, créée par M. Richard Stallman, a pour but d'inciter les développeurs à participer librement. Son préambule, qui sur un plan légal sert à énoncer la volonté des parties et qui est utilisée par le juge pour s'exprimer quand il y a litige sur le fond, édicte que le but de la licence est de préserver les «quatre libertés fondamentales du logiciel libre» :

<sup>3</sup>Le geek ou nerd est un archétype du marginal (scientifique dans le cas du nerd). Non pas au sens de personne qui rejette la société, mais au sens de personne dont l'activité n'est pas attrayante. Le portail communautaire est (<http://www.slashdot.org/>), et le magazine de référence *Wired*

- **la liberté d'utilisation**, tout le monde peut utiliser sans restriction le travail fait par l'auteur ;
- **la liberté d'étudier**, le code peut être regardé et étudié par tous ;
- **liberté de modification**, tout le monde peut modifier le code ;
- **la liberté de copier**, le code doit être fourni avec la même licence à ceux qui l'utilisent, et avec son code source. Il est à noter que cette liberté n'interdit pas la commercialisation, seulement elle autorise la copie sans limite ;

Toute redistribution ou modification doit se faire en respectant ces points. Il est à noter, que cette licence implique que les mélanges de codes ne peuvent se faire qu'avec du code ayant une licence compatible. C'est ce qui a valu à cette licence la qualification de *licence virale*.

### 3.4.2 Influence de la licence sur les pratiques de la communauté Linux

La GPL a été choisie car elle assurait à la communauté la pérennité de son travail. Les autres initiatives de participation à des projets avec des licences différentes ayant été condamnées par la perte de contrôle sur le travail effectué (participation naïve à Unix, procès de BSD et réutilisation massive sans retour du travail). La GPL par sa protection utilisant le copyright s'est révélée efficace. Elle introduit un élément simple : la symétrie entre l'éditeur et l'utilisateur. En effet, cette licence permet que tout utilisateur puisse se substituer à l'éditeur sur les points suivants :

- édition et correction du travail ;
- redistribution du travail ;

Cet aspect important va de pair avec la pratique du *«fork»*, qui assure l'indépendance de l'utilisateur à une autorité centrale. Si l'utilisateur est insatisfait par les pratiques de la communauté qui maintient le projet, il peut dupliquer le travail dans le respect de la licence et le reprendre. Il peut aussi reprendre un projet qui a été abandonné par ses auteurs. Ces objectifs ont été faits directement en réaction des pratiques faites par les entreprises de logiciels dit *«propriétaires»*. Ces logiciels sont distribués sans leur code source, et il est interdit d'essayer de comprendre leur fonctionnement. Ainsi, les utilisateurs de ce genre de logiciels ne sont ni autorisés à corriger les erreurs dans le travail des autres, ni à étudier les spécifications des produits qui leur sont nécessaires <sup>4</sup>

### 3.4.3 Influence de la licence sur la culture

La GPL, de par la personnalité de son fondateur et la force de la FSF en tant que pouvoir de lobbying, influence directement la communauté des développeurs. Quoique la personnalité de Linus Torvalds soit éloignée de la volonté prosélyte et politique de la FSF, la licence est présente en clair-obscur dans le projet Linux à cause des interdépendances avec des projets GNU. Il faut un compilateur et des bibliothèques C pour pouvoir compiler un noyau, et il faut un noyau pour pouvoir faire tourner un compilateur (c'est un peu la dialectique de la poule et de l'oeuf qui s'applique à ce sujet). Ainsi, comme Linux est le premier noyau qui permette à une distribution ( D page 26) GNU de voir le jour, il existe une volonté indéniable de la part de la FSF de mettre la communauté Linux au pas. Elle représente la plus visible des communautés de développeurs. Au sens de Mintzberg, celle-ci est une organisation missionnaire. Elle se retrouve solidaire de la FSF, quant au problème des droits d'auteurs, même si de nombreuses divergences émaillent leurs relations.

## 3.5 Analyse des jeux pratiqués

### 3.5.1 Les asymétries

L'organisation, selon Crozier, ne peut pas être analysée en dehors des luttes de pouvoir. Les luttes de pouvoir sont elles-mêmes repérables par l'existence d'une asymétrie. L'asymétrie est, dans le cadre de l'existence d'une relation entre deux éléments A & B, l'impossibilité de substituer A et B, et à conserver une relation identique (A peut faire faire à B, mais B ne peut pas faire faire à A). Ce qui se traduit en terme de relation par A supérieur à B, à l'instar d'une relation symétrique où A est équivalent à B. Crozier distingue trois types de grandes figures liées aux pouvoirs : l'expert, le redistributeur de l'information, l'édicteur de règles.

### Structure de pouvoir et critères de rationalité

Linus Torvalds se décrit lui-même comme un dictateur bienveillant. C'est effectif, lui seul a longtemps décidé de la validité d'une contribution. En fait, il édicte les critères de rationalité qui conditionnent l'acceptation d'un projet. Cette pratique courante, dans les cercles anglo-saxons, est soutenue par l'exemplarité : la référence est le modèle de soumission qu'a lui-même suivi Linus Torvalds pour poster ses premières contributions. Les critères de rationalité majeurs pour qu'une contribution soit acceptée sont :

<sup>4</sup>Les dernières illustrations en date de cet aspect sont les procès opposant les développeurs d'une solution pour pouvoir lire les DVD sous des plateformes libres (procès DeCSS) , et les éditeurs de ces solutions. Ainsi que l'emprisonnement aux Etats-Unis de Dmitry Zlyakov pour avoir étudié la solution de Adobe permettant de produire des fichiers au format PDF.

- l'utilité du projet (est-ce que cela corrige un défaut, apporte une fonctionnalité nouvelle, ou cela améliore-t-il l'existant ?) ;
- son intégration dans l'existant (essayer de ne pas réinventer la roue) ;
- est-ce que cela fonctionne ? ;
- est-ce éthique (ne pas utiliser du code qui violerait les droits patrimoniaux de son auteur) ? ;

Dans le cadre de cette dictature, la liberté d'expression est totale, et elle ne s'applique que par l'acceptation de tous des discussions, parfois enflammées. Le projet Linux, se voulant avant tout être un projet de «*geeks pour des geeks*», la structure s'apparente à une ad-hocratie opérationnelle au sens de Mintzberg : les parties prenantes, qui sont souvent des experts, fonctionnent par ajustements mutuels.

### L'accès à l'information

L'utilisation de standard pour la communication, la publication systématique du travail, l'archivage des échanges de courriers, assurent à tous, *a priori*, une égalité d'accès à l'information et d'apprentissage de ces pratiques. Or il se trouve qu'aujourd'hui, rien que la correspondance générée par la communauté est de l'ordre de 200 messages par jour. Le point de rupture sur l'accès à l'information se fait en fait sur trois points :

- l'accès à des infrastructures informatiques, et de communication à haut débit : le pré-requis pour accéder à l'information est de pouvoir accéder à Internet dans de bonnes conditions. De ce fait, étant donné la répartition sociale et géographiques des accès à Internet, les populations privilégiées appartiennent aux pays occidentaux dans des couches bien éduquées de la population (particulièrement les universitaires, et les professionnels des technologies de l'information numérique). Dans le commerce, on peut trouver des produits au coût tout à fait raisonnable contenant les documentations, et les sources de Linux. Ces derniers sont cependant moins à jour et moins complet que ce qui est disponible sur Internet.
- la formation : elle nécessite en plus des infrastructures pré-citées, le fait de disposer de suffisamment de temps libre. Ainsi, encore les universitaires et les personnes appartenant à des pays permettant de disposer de congés sabbatiques sont privilégiés ;
- le langage de discussion : c'est l'anglais technique. Il implique que les personnes soient issues de pays anglophones, ou qu'ils aient une éducation suffisante pour pouvoir écrire de manière satisfaisante dans cette langue.
- la connaissance du langage de programmation : il s'agit du langage C. Il est très peu appris en dehors des environnements professionnels et universitaires spécialisés. Néanmoins, c'est un langage d'une conception simple (KISS). Celui utilisé par le projet Linux est proche de la norme ANSI C. De la même manière, les protocoles sont souvent assez techniques, mais les documents de références (RFC ( D page 26)) sont écrits de manière intelligible.

Vu la somme de connaissances générées depuis les débuts, et eu égard aux capacités limitées d'intégration de l'information par l'être humain, il existe une prime *a priori* au premier entrant en terme de capitalisation de l'expérience. Les récentes contributions tendent à prouver que cette prime n'est pas excessive vu l'arrivée de nouveaux venus sur des sujets pointus[20]. Cette communauté est *a priori* exclusive à cause des connaissances techniques nécessaire à l'utilisation des infrastructures[21][22]. Néanmoins les règles tendent d'établir une rétroaction allant dans le sens d'une répartition la plus large possible de l'information.

### Le règne de l'expert

La conception KISS ( D page 26), la documentation fournie, la clarté code source, le respect de normes accessibles à tous permettent une compréhension rapide de l'existant. Certes, il n'existe pas un règne de l'expert technique à proprement parler, en revanche une certaine expertise est nécessaire pour accéder à l'information.

### 3.5.2 Gains et pertes

Les jeux de l'organisation ont comme première finalité de justifier les pouvoirs au sein de l'entreprise [23]. Le projet Linux est formellement dirigé sous le signe de la dictature bienveillante[16]. La règle est claire, tout le monde fait ce qu'il veut tant qu'il se conforme aux règles, leur violation entraîne une peine simple : le fait d'être ignoré. La récompense ultime est d'être reconnu. Un logiciel partageable est un logiciel dont la valeur augmente avec l'usage qui en est fait, et La liberté de copie facilite la propagation du logiciel, et la visibilité de son auteur. Ceci fonctionne un peu comme un théorème. Cet ouvrage d'art auquel participe le développeur augmente sa propre valeur, et plus il est partagé, plus à la fois les auteurs et le produit sont reconnus. Les gains et la finalité des jeux sont orientés vers la reconnaissance, c'est typiquement une problématique de «professionnels»[24]. Le profil gagnant est celui de l'individualiste qui apporte son oeuvre pour se faire reconnaître. Typiquement, pour gagner il faut entreprendre, et la sélection *darwinienne* des projets apporte la validation de la contribution. Cette pratique se rapproche d'une utopie libérale par le caractère d'égalité d'accès aux ressources (ici l'information), et par le caractère de sélection du projet par l'aptitude personnelle d'une part, et par la régulation minimale d'autre part. On peut aussi voir dans cette logique qui s'abstrait d'un environnement économique et politique, une logique d'action (*la propagande par le fait* selon Kropotkine) rappelant les influences anarcho-épistémologiques héritées de la pratique scientifique.

Linux est comme le sport : il peut être envisagé comme l'expression de nombreuses idéologies en contradiction les unes avec les autres. Il peut être vu comme la réussite d'une organisation anarchique par son aspect non-hiérarchique, ou comme une réussite du libéralisme par la sélection darwinienne des contributions.

Un certain nombre de pratiques de la communauté Linux sont en porte-à-faux par rapport à celles l'entreprise :

- le projet n'est pas élaboré à partir d'une étude de marché, dans le cadre de la réalisation d'une stratégie planifiée ;
- il n'y a pas de notion de client, car producteur et consommateur sont les mêmes personnes ;
- le travail est considéré comme un plaisir, donc il est réalisé hors des contraintes de l'entreprise à savoir : le salaire qui oblige au respect de la hiérarchie et des contraintes de temps, et du sujet de travail ;
- la finalité n'est pas l'accession à un poste de responsabilité, qui entraîne la reconnaissance, mais l'accession à la reconnaissance par le travail effectué (qui entraîne souvent de nouvelles responsabilités) ;
- Le contributeur au projet Linux ne travaille pas pour Linux, mais pour lui, et la reconnaissance qu'il escompte vient de son nom plus que du projet.
- il n'y a logique de subordination ad-hoc, ni d'asymétrie de l'information (lutte contre l'opportunisme[7])

Le slogan «*have fun*» est comme un slogan de contestation face à la notion de labeur de la société économique ; l'entreprise se prévaut de rémunérer ceux qui travaillent le plus et la rémunération est proportionnelle au désagrément engendré par la charge de travail (labeur), alors que le projet Linux valorise le plaisir de la création et l'aboutissement de projet qui sont fait dans un esprit ludique. Cette démarche rappelle en de nombreux points les propositions de Fred Brooks [25] dans son essai sur l'ingénierie informatique dans le cadre de l'amélioration des pratiques de l'entreprise. Et bizarrement, au vu du résultat, que produire en se faisant ne donne pas de mauvais résultat.

### 3.5.3 Conclusions

La communauté essaie de placer des règles d'équivalence entre individus participatifs. En fait il y a équivalence entre utilisateurs, éditeurs, et développeurs ... à partir du moment où les parties prenantes (à savoir les personnes soumettant leurs contributions) acceptent le contrat tacite que représentent les protocoles de communication et d'échanges d'information en vigueur.

L'asymétrie majeure ne se fait pas entre les membres du projet, car ils appartiennent *de facto* avant même leur participation à une communauté professionnelle. L'aspect structure missionnaire du projet Linux ne fait que refléter l'intégration à une communauté plus large dite «logiciel libre», qui correspond à une défense du mode coopératif menacé par des pratiques d'accaparement supposée ou réelle de la connaissance. Ces pratiques menace le fonctionnement et l'existence même de cette communauté, c'est ainsi que la communauté est partie prenante dans la lutte contre les brevets logiciels[26].

La communauté Linux a pour vocation première de produire un bien immatériel hautement technique et toutes les règles sont faites pour faciliter l'échange d'information technique, et encourager les initiatives permettant l'apport d'innovations fonctionnelles.

Le développeurs Linux voient dans l'accomplissement du projet général la reconnaissance de leur projet personnel. L'ensemble de règle de participation individuelles définit un cadre augmentant le bien commun.

La véritable difficulté est au final de rentrer dans la communauté. Cela se fait par un apprentissage voire une initiation. Cet apprentissage nécessite à la base trois ingrédients :

- du temps ;
- un accès à Internet ;
- être suffisamment «lettré» en anglais technique pour savoir trouver les informations ;
- et souvent, il est accéléré par l'aide d'une personne appartenant déjà à une des communautés du logiciel libre.

# Chapitre 4

## Conclusions

### 4.1 Linux et l'innovation : un cercle vertueux

S'inscrivant sur le plan social, il s'agit d'un projet universaliste ayant pour vocation la reconnaissance de l'individu, et le libre accès à l'information, et l'égalité des chances : la barrière des diplômes et de l'expérience professionnelle utilisées en entreprise a été reconnue pour être arbitraire[27]<sup>1</sup>. La reconnaissance par les pairs se substitue à une autorité centrale de référence le cadre de l'évaluation des personnes comme peut l'être un professeur pour ses élèves<sup>2</sup>. Ainsi, la créativité peut elle s'exprimer car elle se mesure par rapport aux résultats. Sinon, dans le cadre strict de la production, la libre publication du code augmente sa valeur d'usage, et la renommée du logiciel augmente la renommée de ses développeurs. Il s'agit d'un jeu définissant un cercle vertueux de l'augmentation de la valeur personnelle par la valeur du produit. La finalité de la communauté et la motivation des ces parties constituent un ensemble solidaire dans la création de valeur.

### 4.2 Linux et l'entreprise

La première partie met l'accent sur l'importance de la notion de bureaucratie forte et de la planification au sein de l'entreprise. Linux se base sur la production de résultat car il fonctionne dans une optique contraire des règles bureaucratiques. C'est à dire le refus de liens de subordination et d'asymétrie de l'information. Le contrat de travail permet d'avoir des individus qui exécutent des actions qu'ils ne décident pas en l'absence d'informations suffisantes. La clef du succès du projet Linux est le système participatif basé sur la confiance des pairs, et la transparence. Ainsi, là où le secret industriel fait la force de produits tels que Coca Cola, dont la recette est l'un des secrets industriels les mieux gardés, la force de Linux est la possibilité d'accéder librement à l'ensemble de ses sources, et à la communauté elle-même.

Là où le fonctionnement bureaucratique fixe rigidement la manière de faire, et les chemins de l'information, l'organisation Linux, elle, ne fixe sa contingence que sur les résultats, et fait évoluer ses pratiques et ses méthodes. Elle atteint quatre qualités fondamentales que recherchent les entreprises :

- l'innovation, par le développement de la créativité et de l'entrepreneuriat ;
- la réactivité, par l'utilisation de règles de communication universelles[28] ;
- la qualité, par l'écoute et le temps de développement qu'elle a choisis.
- l'apprentissage, par l'utilisation de règles souples de fonctionnement.

La problématique du temps (et donc de la planification) conditionne la réussite des projets[25], à ce titre Linux, en prenant comme principe de donner du temps au temps, apporte une certaine qualité à ses produits qui est de prime abord incompatible avec le monde de l'entreprise et de ses échéances.

### 4.3 Une aberration organisationnelle ?

La structure du projet Linux est organisée. Très organisée même, car la minimalité des structures et des règles permet une bonne transmission des stratégies et valeurs de la structure. De par sa réussite, ce système a fait ses preuves. De par sa légèreté il peut être considéré comme un système efficace c'est-à-dire ayant un bon rendement. Le surcoût en temps induit par les duplications d'effort semble compenser le temps qui n'est pas passé à effectuer une coordination formelle forte. La duplication de projet en augmentant les alternatives semble aussi augmenter la qualité. Quand il y a plus d'un choix, il est possible de faire le bon choix. Le projet a aussi montré sa capacité à faire collaborer sur un projet des individus fortement individualistes en exploitant leur créativité, et à leur faire assumer pour certains des responsabilités

<sup>1</sup>Cette étude de 1968 montre que la productivité en entreprise est indépendante du diplôme, et de l'expérience (pour un programmeur en activité en entreprise depuis 2 ans).

<sup>2</sup>Ce comportement se retrouve dans le domaine de la cryptographie, où ces mêmes développeurs ont préféré un système de certification par les pairs basé sur les «cercles de confiance»(*ring of trust*) plutôt que sur une autorité centrale publique ou commerciale de certification (comme les dépositaires de clés du ministère de l'Intérieur en France, ou de la société Verisign dans le domaine du commerce électronique.)

communautaires. L'individualisme ne peut plus être considéré comme un frein au travail d'équipe. Bref, malgré son qualificatif de «*bazar*», cette organisation est structurée. En fait, plutôt qu'une aberration, c'est une organisation qui a expérimentée de nouvelles méthodes.

#### **4.4 Des mécanismes d'acculturation sont-ils envisageables entre des structures d'entreprises classiques et la communauté Linux ?**

Les pratiques et les outils utilisés dans le cadre du projet Linux sont hérités de l'entreprise. Néanmoins, leur mise en oeuvre et la disparition de la hiérarchie sont en contre-pied avec les pratiques des entreprises qui ont des formes bureaucratiques. Là où une entreprise apporte à ses salariés le comment-faire sous la forme de la standardisation des procédures et le savoir-être, le projet Linux apporte à ses contributeurs le savoir-réussir. Ce dernier peut se résumer à apporter une contribution fonctionnelle.

Les pratiques de non-diffusion de l'information (brevet, NDA), ainsi que les pratiques productivistes (édiction de dates butoir), sont incompatibles avec le projet Linux en particulier, et les projets logiciels libres en général. Les micro-sociétés comme certaines structures autonomes, et département de R&D semblent les plus adaptés à dialoguer avec la communauté Linux. Cela peut s'expliquer comme suit : ces structures sont dédiées à la résolution de problèmes non-standard nécessitant une approche de réutilisation de composants standards et d'amélioration ponctuelle. Les organisations *ad-hocratiques* fonctionnent plutôt par ajustement mutuel comme la communauté Linux. L'acculturation s'imagine plus facilement avec des organisations qui lui sont proches. La planification doit se faire de façon harmonieuse par rapport aux conventions et à la culture Linux. La structure pour obtenir des résultats doit pouvoir fixer des objectifs raisonnables pour l'entreprise, tout tout en respectant les conventions de la communauté. Ceci n'est possible que dans des structures pouvant se déjà se permettre une certaine souplesse par rapports à ses propres objectifs.

L'adhésion des contributeurs Linux aux conventions et aux objectifs est non contrainte, bien que les actions des individus soit libres, elles le sont dans des conditions non librement déterminées. La survie de l'entreprise est contraintes par des critères de résultats et de délais de livraisons contractuellement déterminés entre parties. Il semble difficile d'imaginer une entreprise qui adopterait pour l'ensemble de sa structure le même fonctionnement que le logiciel libre : la communauté Linux, vu son mode de fonctionnement, a des résultats imprévisible et à ce titre difficile à quantifier. Il me semble non trivial de vendre une prestation dont qu'on ne peut évaluer. Utiliser les méthodes du logiciel libre ponctuellement en entreprise pour des communautés de professionnels semble réalisable. Par contre, faire fonctionner toute une entreprise selon ces règles semble quelque peu risqué.

#### **4.5 Linux et la société : cyber-vigilance et *hactivism***

Les associations comme ATTAC, ou les magazines de défense des libertés civiles comme le réseau Voltaire ont publiquement adoptés les logiciels libres comme plate-forme de diffusion de leurs informations. On peut parler avec l'émergence des projets logiciel libre d'une contre-culture citoyenne appelée *hactivism*[29] [30]. Les hactivistes peuvent être définis comme les vigies citoyennes et agissantes qui observent le domaine des nouvelles technologies de l'information. En parallèle, on voit émerger d'autres mouvements qui à travers la production d'outils et de méthodes collaboratives essaie d'élaborer un bien (être) commun. Cette culture n'a pas pour but la disparition du secteur privé, mais son évolution vers des pratiques plus respectueuses de l'intérêt commun. Cette tendance s'exprime notamment à travers des problématiques de coopération, et de développement durable[31].

**Deuxième partie**

**Bibliographie**

# Bibliographie

- [1] Site de la chaire "Développement des Systèmes d'Organisation" du CNAM. <http://www.cnam.fr/depts/te/dso/>.
- [2] P. Lemaître. *Organisation du travail et de l'entreprise, Cours A*. CNAM/Media, 2001/2002.
- [3] traduction de Nat Makarevitch Eric S Raymond. Distinction GNU / Linux. [http://www.linux-france.org/article/these/linux\\_et\\_gnu/linux\\_et\\_gnu.html](http://www.linux-france.org/article/these/linux_et_gnu/linux_et_gnu.html), 1998.
- [4] Freshmeat. <http://www.freshmeat.net/>. Site de recensement des projets logiciel libre publiés.
- [5] Eric S Raymond. *La Cathédrale et le Bazar*. O'Reilly Edition, 1998. Recueil papier des articles publiés sur internet.
- [6] The Nature of Firm. *economica*, 1937.
- [7] Olivier E. Williamson. *The Mechanisms of Governance*. oxford press, 1996.
- [8] Azariadis. Implicit contracts and unemployment equilibria. *economica*, 1975.
- [9] David Kreps. *Perspectives on Positive Political Economy*, chapter Corporate Culture and Economic Theory. Cambridge University press, 1990.
- [10] Erhard Friedberg Michel Crozier. *L'acteur et le système, Les contraintes de l'action collective*. Editions du Seuil, 1981. L'analyse systémique appliquée aux organisations.
- [11] Maurice Thévenet. *La culture d'entreprise*. Paris, PUF, col. Que sais je ?, 1994.
- [12] Archives de la liste de diffusion électronique pour le projet Linux. <http://kt.linuxcare.com/kernel-traffic/>.
- [13] Étude sur les motivations des développeurs linux. <http://www.psychologie.uni-kiel.de/linux-study/>.
- [14] M. Weber. *L'éthique protestante et l'esprit du capitalisme*. Plon, Paris, 1994. <http://www.ao.qc.ca/archives/nos/4-2refu/capitalisme.html>.
- [15] Andrew Tannenbaum. *Operating System. Designs and implementations*. Prentice Hall, 1987.
- [16] David Diamonds Linus Torvalds. *Just For Fun*. Texere, 2001. Biographie de Linus Torvalds.
- [17] Eric S Raymond. *La Cathédrale et le Bazar*. O'Reilly Edition, 1998. Description du hacking comme activité de pionnier aux frontières de la connaissance informatique.
- [18] Un essai sur les enjeux du conflit Linus Torvalds/Bill Gates. <http://www.volle.com/opinion/billinus.htm>.
- [19] Histoire de l'émergence de BSD . <http://www.oreilly.com/catalog/opensources/book/kirkmck.html>.
- [20] Annonce d'une nouvelle implémentation d'une fonction pointue par Ingo Molnar. <http://www.uwsg.indiana.edu/hypertext/linux/kernel/0101.3/0930.html>.
- [21] Site de développement collaboratif autour d'une thématique logiciel libre. <http://www.tuxfamily.org/>.
- [22] Ferme de développement pour projet logiciel libre. <http://savannah.gnu.org/>.
- [23] Erhard Friedberg Michel Crozier. *L'acteur et le système, Les contraintes de l'action collective*, chapter Le pouvoir comme fondement de l'action organisée. Editions du Seuil, 1981. L'analyse systémique appliquée aux organisations.
- [24] Karl Eric Sveiby. Knowledge Strategy. *International Review of Strategic Management*, 3, 1992. The knowledge company : strategy formulation in knowledge-intensive industries, <http://www.sveiby.com/articles/KnowledgeStrategy.htm>.
- [25] Frederic Brooks. *The Mythical Man-Month*. Addison Wesley, 1975. Essai sur l'ingénierie logiciel.
- [26] Site de lutte contre les brevets logiciels en Europe. <http://www.eurolinux.org/>.
- [27] Grant Sackman, Erikson. Exploratory experimental studies comparing online and offline programming performance, Janvier 1968.
- [28] Traduction française de la nétiquette. <http://www.sri.ucl.ac.be/SRI/rfc1855.fr.html>.
- [29] samizdat. <http://hns.samizdat.net/>. Site de news hacktivist «don't hate the media, be the media».
- [30] Big Brother Awards. <http://www.bigbrotherawards.eu.org/2001/nomines.html>. site des bigs brother awards, vigilance sur les atteintes aux libertés civiles.
- [31] Site pour la coopération nord-sud par les moyens numériques, et la francophone . <http://www.ynternet.org/>.

- [32] <http://www.gnu.org/philosophy/>. description du logiciel libre.
- [33] Description du projet GNU. <http://www.gnu.org/>.
- [34] Association Bordelaise des Utilisateurs de Linux. *Rencontres mondiales du logiciel libre*, 2001. Communication officielle de la FSF.
- [35] Eric S Raymond. <http://www.linux-france.org/article/these/cathedrale-bazar/>, pages Le chaudron magique magic-cauldron-fr\_monoblock.html. 1998. Les modèles économiques basés sur le logiciel Open Source.
- [36] Frederic Brooks. *The Mythical Man-Month*, chapter Aristocracy, Democracy, and Sytem Design. Addison Wesley, 1975. Essai sur l'ingénierie logiciel.
- [37] Coût en entreprise d'une distribution GNU/Linux. <http://people.debian.org/jgb/debian-counting/counting-potatoes/>.
- [38] David Diamonds Linus Torvalds. *Just For Fun*, page 73. Texere, 2001. Biographie de Linus Torvalds (début de Linux comme système d'exploitation).
- [39] Ensemble des articles, traduction et travaux juridiques concernant la GPL. <http://france.fsfeurope.org/gpl/gpl.fr.html>.
- [40] Traduction de Sébastien Blondeel Eric S Raymond. *Une brève histoire des hackers*, pages fr-a\_brief\_history\_of\_hackerdom\_monoblock.html. O'Reilly, 1998.
- [41] Règles de fonctionnement de la liste de diffusion électronique pour le projet Linux. <http://www.tux.org/lkml>, 1992.
- [42] Ed Foster. <http://www.infoworld.com/cgi-bin/displayTC.pl?97poy.suppl.htm>, page Meilleur support technique de l'année 1998. IDG group press, 1998. Meilleur support Technique 1997.
- [43] Paul 'Rusty' Russel. <http://netfilter.samba.org/unreliables-guides/fr/>, chapter netfilter-hacking-HOWTO-7.html. Difficulté d'élaboration de la dernière version du noyau Linux.
- [44] Consortium Unix. *Monterey*, 1998.
- [45] Traduction de Sébastien Blondeel Entrevue de Rishab Aiyer Ghosh. *Qu'est-ce qui motive les développeurs de logiciels libres ?*, pages [http://www.linux-france.org/article/these/interview/torvalds/fr-lt\\_first\\_monday.199803.html](http://www.linux-france.org/article/these/interview/torvalds/fr-lt_first_monday.199803.html). 1998.
- [46] David Diamonds Linus Torvalds. *Just For Fun*, page 169. Texere, 2001. Biographie de Linus Torvalds (Linus Torvalds chez Transmeta).
- [47] Linux World News. Full scale flame-war between Linux and GGI project, GGI leaves Linux project. <http://lwnet/1998/0402>, 1998.

## **Troisième partie**

### **Annexes**

# Annexe A

## Linux : qu'est ce ?

### A.1 Un Unix

Un Unix est un système d'exploitation ( D page 27) conforme à des normes strictes. Conçu par Richie et Thompson au début des années 1970, il avait pour vocation :

- d'être simple ( D page 26),
- d'être portable, c'est-à-dire que chaque fabricant de matériel pouvait concevoir son Unix qui lui était propre, mais que les applications n'en étaient pas dépendantes. C'est ce qu'on appelle la portabilité.

Les normes, si elles sont respectées, assurent la portabilité et simplifient le développement des applications. Linux lors de sa conception a été conçu en respectant l'ensemble des normes en vigueur concernant le système, les encodages de polices, les couches réseaux...

C'est ainsi, que le consortium Unix[44], composé de la plupart des éditeurs commerciaux (HP, IBM, Compaq, SGI ...), a décidé d'annoncer sa conformité avec les standards. Cette annonce a changé Linux du statut de système d'exploitation «*de bidouilleurs*» à celui de système d'exploitation utilisable en environnement professionnel.

### A.2 Le projet de Linus Torvalds

Linux (du nom de travail de Linus Torvalds[16]) est apparu comme la réalisation d'un projet qui correspondait à une envie de ce dernier de faire un système d'exploitation.

### A.3 Une structure virtuelle

La première version fut développée par Linus Torvalds dans sa chambre. Ensuite, à partir du moment où son travail fut annoncé, un professeur de l'université d'Helsinki (Finlande) lui proposa un espace de stockage accessible via Internet pour permettre à d'autres de le télécharger. Les développeurs collaborent sur Internet en utilisant juste le courrier électronique et un espace de stockage permettant de mettre le système à jour. Cette structure n'a rien de formelle et est basée sur la coopération d'individus qui se rencontrent quasi-exclusivement de cette manière.

### A.4 Le coeur d'un système hétérogène cohérent

On désigne par Linux le noyau de l'Unix. Le noyau du système d'exploitation seul n'est pas suffisant pour être opérationnel. Il faut lui adjoindre des outils d'administration et des outils logiciels pour qu'il soit utilisable. On désigne par *GNU/Linux* l'agrégat constitué du noyau et des logiciels nécessaires pour avoir un système utilisable par un être humain. La plupart des briques de base ont été développées dans le cadre du projet GNU ( D page 26).

Communément, dans les publications grand public on fait l'amalgame entre Linux et *GNU/Linux*. La discussion est en fait idéologique[34][3]. En conclusion, il est important de distinguer :

- **la communauté Linux**, l'ensemble des parties prenantes qui partagent une culture et des usages communs et qui s'inscrit dans un environnement plus large ;
- **le projet Linux**, qui est l'ensemble des pratiques, et des méthodes utilisées par les individus en vue de pouvoir réaliser un travail créatif ;
- **Linux** qui est le résultat obtenu par la communauté dans le cadre du projet, et qui lui est un noyau, la réalisation.

## **Annexe B**

# **Historique du logiciel libre, et de Linux**

Se référer à <http://projet.unix.free.fr/historique.htm>. Je me suis aperçu que d'autres l'avaient très bien fait.

# Annexe C

## Les règles de la communauté Linux

L'environnement modèle les règles du jeu. Les influences majeures à prendre en compte sont :

- l'environnement universitaire dont sont issus la plupart des développeurs du coeur de la communauté,
- Internet qui véhicule un certain nombre de règles quant à la façon de dialoguer.

Les règles sont accessibles essentiellement à deux endroits :

- par la nétiquette (RFC 1855[28]), ou pratique de l'Internet poli,
- par la description qu'en fait Eric S. Raymond[5].

Ces règles qui sont vraies sur le projet Linux sont vraies pour la plupart des projets logiciel libre. En effet, certains projets sont développés dans le mode «*cathédrale*», où pour conserver «*l'intégrité*» du logiciel, le projet se développe entre personnes qui se connaissent (projet Emacs), et éventuellement les développeurs sont intégrés par cooptation (projet XFree 86). Ici nous traitons du mode «*bazar*».

### C.1 Lancer le projet

#### C.1.1 Genèse

Commencer par avoir l'idée de quelque chose :

- d'amusant,
- et d'utile pour soi.

Vérifier que le même projet n'existe pas, car le «*fork*»( D page 26) est considéré comme une duplication inutile d'effort. Il est préférable de parler de son projet à des personnes ayant des projets similaires. Cette règle implique que la personne se soit documenter sérieusement. S'auto-documenter est une des règles de base qui va de pair, dans le cadre d'une conduite cohérente avec le fait de documenter.

S'inspirer au maximum de projet voisins.

Commencer par faire un prototype selon la règle *plan to throw one away, you will do it anyway*[25].

L'usage est de concevoir selon une conception KISS ( D page 26), et de se concentrer sur les structures de données plus que sur le programme[25].

S'il est question d'échange de données, chercher dès le début du projet à respecter les standards et les RFC ( D page 26) afin d'être le plus ouvert sur les autres systèmes d'exploitation. Ne jamais perdre d'informations (compatibilité) sauf lorsque des contraintes matérielles l'y obligent. Et ne pas perturber les flots de données.

Pour un projet de sécurité, ne pas se baser sur le secret mais sur une conception efficace (ce n'est pas parce que l'on ignore la conception de votre serrure que l'on ne pourra pas la forcer.)

#### C.1.2 Publication

Le projet est publié selon la formule *release early, release often*, plus le code est publié tôt, plus la dynamique a de chance de se produire, et il est plus valorisant pour des personnes d'apporter leurs idées avant que le projet ne soit trop abouti.

La publication doit se faire par les moyens d'une infrastructure publique accessible à tous.

Une licence doit être choisie pour rester en accord avec l'utilisation, les personnes qui le développent, et les autres projets sur lesquels le projet s'appuie. Pour obtenir le concours de la communauté, l'utilisation d'une licence compatible logiciel libre est conseillée[32].

Comme le projet a vocation à être étudié par d'autres, il est chaudement conseillé d'écrire son code proprement et de le documenter, c'est aussi une question de réputation. Le code reste la première source de documentation.

### C.1.3 annoncer

Il existe, si votre projet est utile, toujours une communauté de personne désirant l'utiliser, et y participer. Il est important d'identifier cette communauté et de lui adresser un message via les forums adéquats. Après, le résultat dépend de la pertinence de votre projet et de votre qualité d'animateur ...

## C.2 Maintenance du projet et participation à un projet

Lancer le projet ou le sous-projet est une partie importante, car le non-respect d'une partie de ses règles peut amener à la désaffection des développeurs/utilisateurs potentiels. C'est ce qui est arrivé à des sociétés comme *Netscape*, *Apple*, *Zend*, lors du lancement de leurs projets Open Source.

### C.2.1 Soumettre des contributions

À la question comment puis-je contribuer, la réponse des développeurs est invariablement, montrez-moi votre contribution. Les projets logiciels libres comme Linux fonctionnent avec la règle comme quoi il est inutile de diriger les personnes. Donc, il n'y a pas d'affectation de tâches. Le pendant, est qu'il n'y a pas non plus de recherche afin d'éliminer les doublons d'effort. A priori les développeurs ont conscience d'un certain tabou face au «fork» (duplication d'un projet existant), et donc, même si une contribution est similaire à une autre existante, on laisse les contributions être faites après avoir averti de l'existence d'un projet similaire. La plupart du temps :

- soit les projets se révèlent complémentaires et divergent *in fine* ;
- soit chaque projet apporte un point original, et les approches complémentaires sont fusionnées en un projet unique, on laisse la sélection darwinienne choisir le projet qui devient le projet principal.

### C.2.2 Intégrer des contributions

Avant de rentrer dans ce sujet, il faut noter que la conception a été faite de manière à ce que les sous-projets puissent être modulaires, pour concevoir ou développer une sous-partie d'un noyau, il n'est pas nécessaire de connaître l'intégralité de ce qui a été conçu par les autres, ce qui se reflète dans la structuration des équipes.

Dans le cadre du projet Linux, l'intégration des contributions se fait sous l'avis soit

- de Linus Torvalds, ou Alan Cox si il s'agit d'un nouvel élément majeur apporté au projet ;
- du chef du sous-projet, si cela appartient à un sous projet du noyau (gestion de périphériques spécifiques, système de fichiers).

Il suffit d'envoyer sa contribution, elle est examinée. La talent d'un mainteneur comme Linux, et de savoir mettre les gens en confiance, et de les aider le cas échéant. Le contributeur et l'utilisateur sont les deux principales ressources d'un projet avant le code lui même et sa documentation, il est important de savoir les garder.

### C.2.3 Documenter

La documentation est un point essentiel : pour que le travail puisse être apprécié à sa juste valeur, les autres doivent pouvoir le comprendre et l'utiliser, l'idée est aussi de pouvoir utiliser le travail des autres. La documentation recouvre les domaines suivants :

- le code source disponible est la première source de documentation : il doit être claire, cohérent, et contenir si nécessaire des commentaires intégrés directement dans le code ;
- les modifications faites doivent pouvoir être suivies, ainsi tout travail s'accompagne de la mise à jour du fichier contenant les modifications faites (appelé *changelog*) ;
- si un point mérite de l'attention parce qu'il est particulier, un fichier au nom évocateur est placé dans le répertoire des documentations ;
- si un code est fait à partir d'une base classique, un squelette peut servir de supplément à la documentation.
- le travail de documentation et de traduction est une contribution placée sur un plan d'égalité avec celle de l'écriture de code ;
- si quelqu'un s'exprime sur un sujet, il a d'autant plus de légitimité qu'il apporte une contribution : c'est **la prime à l'action** ;

La documentation facilite la prise en main du projet par les nouveaux contributeurs.

### C.2.4 Discuter

Les règles sont simples : il faut respecter la netiquette[28]. Celle-ci indique :

- des messages en format texte pour éviter aux personnes ayant des accès Internet lents ou limités de perdre du temps ;

- la tailles des messages ne doit pas être trop grandes pour qu’ils restent lisibles, donc on ne cite que les parties nécessaires dans une réponse ;
- le respect des règles élémentaires de politesse ;
- on est poli avec les nouveaux arrivants, même si les questions tombent sous le sens ;
- d’un autre côté, il est malvenu de poser des questions sans avoir fait un effort d’auto-documentation sinon on risque un injonction de type RTFM ( D page 27).
- il faut respecter le sujet de discussion du forum : les exemples dans le cas d’un forum technique, les autres sujets, sauf s’ils concernent directement la communauté, ne sont pas acceptés *a priori*.

### C.3 Rapport avec les utilisateurs

Autant il arrive entre développeurs d’avoir des propos enflammés (appelés «*flamewars*»), autant l’utilisateur est traité avec plus de respect. En effet, plus il y a d’utilisateurs, plus le projet a de chance de s’améliorer <sup>1</sup>. Incidemment, la valeur d’usage du produit augmente elle aussi ; plus le produit est utilisé, plus il a de chance de s’imposer. Le respect du nouveau venu est donc de mise.

### C.4 Travail en groupe

Le projet Linux est constitué comme la plupart des projets logiciels libres de l’agrégation de *contributions*. La contribution est l’effort d’un personne apporté à tous. Pour ce qui est du travail, le noyau Linux favorise le travail individuel. La collaboration se fait au niveau du partage de l’information. Le seul point commun des parties prenantes sont les discussions. Cette communauté a d’ailleurs reçu régulièrement des prix pour le meilleur support technique en informatique [42] (en concurrence avec des entreprises).

### C.5 Autorité

L’autorité appartient au chef de sous-projet d’abord, et ensuite au développeur de coeur. Ceci reflète l’une des pratiques ad-hocratique préconisée par Fred Brooks [36]. Elle se limite à accepter l’intégration des contributions dans la branche officielle du projet, et à conserver au cours des développements *l’intégrité conceptuelle* du projet.

Toute personne se sentant en opposition avec l’équipe principale peut lancer un projet indépendant (fork ( D page 26)). Ceci est arrivé à plusieurs reprises [47].

Formellement, le projet Linux est basé sur un fonctionnement tyrannique. Avoir un produit utilisable nécessite à la fois d’avoir une autorité pour centraliser les contributions, mais aussi d’avoir des contributions en nombre suffisant pour garder une dynamique. L’autoritarisme est par conséquent assez souple.

La partie la plus gratifiante d’un projet étant considérée comme étant de prendre part à la réalisation, l’autorité *per se* n’est pas recherchée. En effet, je rappelle que les listes de diffusion techniques peuvent aboutir à un trafic d’une centaine de messages par jour. Par ailleurs les règles de comportement sont accessibles par tous[41]. Le fonctionnement du noyau est plus basé sur le respect de règles intériorisées par les parties prenantes, que sur la figure d’une autorité centrale.

#### C.5.1 Récompenses

**Citation** Tout auteur dont la contribution est intégrée au au code source d’un projet indique, son nom, son adresse email, et si il y a lieu le nom de l’organisation qui le mécène<sup>2</sup>

**Reconnaissance par les pairs** Le fait d’avoir son travail intégré dans le projet permet d’acquérir une légitimité dans le cadre du projet. Par exemple, on fait appel aux personnes qui ont contribué lorsqu’il est nécessaire d’exprimer son avis sur le sujet, même lors de conférences internationales. Les participations aux conférences sont faites à titre gracieux la plupart du temps, il est aussi attendu que le tarif d’entrée soit lui aussi gratuit.

La réputation de la personne est directement basée sur ses contributions et leur qualité. Ainsi, un développeur sera fier que son travail soit intégré dans un autre projet. Car à chaque fois que son travail original est utilisé, sa réputation augmente.

La reconnaissance de la communauté se traduit *de facto* par la reconnaissance d’une expertise technique indépendante du diplôme et de l’expérience de son détenteur.

#### C.5.2 Pénalités

**RTFM ( D page 27) “Relis Ton Fabuleux Manuel”** Cette réplique est rarement utilisée ... en public. Elle est habituellement utilisée en dernier recours après avoir indiqué les références documentaires. Elle manifeste l’énervement de

<sup>1</sup>«*the more eyeballs will meet the code, the more bugs are likely to be found*»(Loi de Linus)

<sup>2</sup>le plus souvent l’entreprise dans laquelle l’auteur travaille et qui l’a laissé faire sans réclamer les droits sur son travail.

l'interlocuteur, et invite à consulter les ressources documentaires. Si la personne persiste, elle risque d'être ignorée dans le futur.

**Death penalty (*peine de mort*)** Cette peine date de l'époque des tous premiers forums de discussion. Lorsque un interlocuteur outrepassait systématiquement les règles de correction (netiquette, ou piratage), cette personnes, ou l'organisation dont elle était issue, étaient considérées comme indésirables. Elles étaient automatiquement ignorées, en fait, mise à l'index. Cette méthode est encore utilisée aujourd'hui dans la lutte contre les mails non sollicités (*pourriels en français ou spams en anglais*).

**Mauvaise réputation** "*reputation die, and grow hard*" De même qu'Internet permet de pouvoir atteindre une notoriété, il ne pardonne pas ; le nom de la personne reste longtemps référencé sur les sites webs accessibles à a partir de moteur de recherche. Or, les forums de discussion par email sont archivés par l'intermédiaire de sites web[12]. La réputation est importante pour pouvoir assurer la viabilité d'un projet et être rejoint notamment par d'autres développeurs pour pouvoir le faire vivre, voire tout simplement pour être crédible.

Les sanctions ont l'air bénignes. Mais elles sont centrées sur les valeurs de ce groupes : la réputation/la reconnaissance par les pairs.

## C.6 Les développeurs Linux face à l'argent

Il y a eu une évolution flagrante de l'attitude de la communauté sur ce point. Avant les débuts de l'avènement de Linux, la plupart des développeurs effectuaient leur développement sur du temps dont ils avaient la maîtrise. À ce titre, il était considéré comme logique que la date limite de rendu disparaisse ; «la programmation c'est comme la cuisine, il faut un certain temps incompressible pour obtenir un résultat correct» [25].

Les développeurs individuels considèrent que Linux est gratuit, et qu'à ce titre, il est ridicule que les gens paient, il est aussi considéré que les pressions apportées par le travail en entreprise sont incompatibles avec le développement de qualité [43]. C'est-à-dire que le consommateur consommerait en fonction de sa perception et que le travail en entreprise est considéré comme cosmétique. Donc, pour faire du bon travail, il suffit de le faire hors du cadre de l'entreprise. En fait, dans la plupart des cas, le salaire pour un développeur Linux est assez souvent négocié selon les conditions du mécénat :

- le mécène devient cosignataire du travail ;
- le développeur a une plage de temps libre pour s'occuper de ce qui lui tient à coeur [16], et il ne doit pas avoir de compte de rendre à ce sujet.

Depuis la reconnaissance par les professionnels de Linux, une nouvelle tendance est apparue : certaines parties prenantes (éditeurs de matériels) fournissent eux-mêmes du code, pour éviter que les développeurs extérieurs ne se désintéressent à leur architecture. Ainsi, sont apparus des développeurs dédiés. De plus des sociétés ont maintenant bâti leur modèle économique sur Linux (*Mandrakesoft, Red Hat, ...*), et l'entreprise/partie prenante du projet a rendu la limite plus floue.

À partir du moment où Linux est arrivé sous les feux de la rampe, la politique par rapport au délai a changé. Linus Torvalds a imposé à la communauté une date de publication par le noyau (début 2000) qui n'a jamais été atteinte, et qui a entraîné un certain nombre de dysfonctionnements aussi bien sur le plan technique que celui de la motivation.

Le fait que les développeurs aient souvent refusé des salaires pour développer ce qui leur plaît ne signifie pas qu'ils refusent d'être payés, et qu'ils ne veulent pas faire d'argent avec. *Le terme exact ce n'est pas de faire de l'argent avec mais autour*. Une personne qui est capable de programmer avec brio possède quelque chose qui lui est inaliénable : sa créativité. Et c'est ce qu'il est prêt à louer.

La plupart de ceux que j'ai pu rencontrer en France ont préféré accepter des salaires moindres par rapport à leurs qualifications. L'employé subit moins de pression de son employeur ainsi, il possède une marge de manoeuvre, lui permettant de négocier le fait de travailler sur un sujet qui présente un intérêt personnel.

# Annexe D

## Vocabulaire

La plupart des termes employés sont anglo-saxons. Les termes proposés par l'académie française tombe assez souvent à plat ("babillard" pour *chat*, "attaque au lance-flamme" pour *flamewar*, "finaud" pour *hacker*). Aussi, à moins que les termes aient émergé au fur et à mesure des usages de la communauté francophone, je choisis d'utiliser les termes anglo-saxons.

**AT&T** AT&T est une entreprise américaine de télécommunication. Elle était notamment réputée dans les années 1980 pour son laboratoire de recherche. En 1984, le Département de Justice Américain ouvre une enquête selon les termes de la loi antitrust. A la suite d'un accord à l'amiable, AT&T accepte d'être séparé en 8 entités.

**BSD** Berkeley Source Distribution[19]. Cet Unix développé en s'inspirant de l'Unix de AT&T, est celui que l'université mis à disposition de tous. Sa licence est beaucoup moins restrictive que celle d Linux. BSD a inspiré et a été utilisée comme base pour de nombreux produits (les premières versions de SunOS sont une version dérivée de BSD).

**Code source** Les sources d'un logiciel sont les fichiers écrits dans un format compréhensible pour un humain, qui permettent de générer le logiciel. Le logiciel est obtenu à partir de la traduction de ces fichiers en langage compréhensible uniquement par la machine. Cette opération s'appelle compilation ou interprétation. Les fichiers sources sont aussi appelés code source.

**Distribution** Dans l'optique des logiciels libres, une distribution est l'agrégat cohérent d'un noyau avec les outils nécessaires à la création d'un système d'exploitation, ainsi que des logiciels qui sont spécifiquement intégrés.

**Free Software Foundation** La FSF est une entité créée par M. Richard Stallman pour développer l'usage du logiciel libre. Cette organisation de lobbying a aussi en charge la défense des intérêts des logiciels ayant décidé de se conformer à la Licence Public Générale, et aux pratiques de la FSF.

**Fork** Le fork est la possibilité pour un développeur de reprendre un projet existant, et d'en faire une autre version, selon sa vision. La plupart du temps, il y a fork après une dissension entre des personnes prenant part au même projet.

**Geek** Le mot «*geek*» désigne en général une forme d'inadaptation sociale qui tire son origine de l'incapacité à intégrer une communauté. Une variante, le «*nerd*» désigne particulièrement les personnes suivant ou ayant suivi un cursus scientifique.

**GNU** Acronyme récursif signifiant GNU's Not Unix (GNU n'est pas Unix)[33]. Le projet GNU a été lancé en 1984. L'objectif est de bâtir un système d'exploitation basé sur des logiciels libres conforme aux normes Unix. Aujourd'hui des variantes du projet utilisant le noyau Linux sont couramment utilisées. Ces variantes sont appelées GNU/Linux.

**Hacker** Plutôt que de paraphraser les définitions existantes [40]. Je résumerais en disant que le mot hack est issu d'une onomatopée désignant le bruit que font les touches d'un clavier quand elles sont enfoncées. Le mot hacker désigne donc en fait un programmeur acharné. La définition fournie en référence doit être modérée. Eric S. Raymond considère cette pratique comme exclusivement sise aux États-Unis, et gomme l'existence de mouvements parallèles en Europe et en Asie. L'Académie Française propose la terminologie "finaud", peu usitée.

**KISS** Keep It Simple Stupid. Philosophie de conception de logiciels basée sur l'idée que le logiciel est d'autant plus simple à maintenir que les gens peuvent l'appréhender facilement. Les efforts sont donc fait sur la conception qui doit être raffinée, c'est à dire simple.

**Licence Publique Générale (LGP)** [39] (GPL General Public License) Cette licence a pour objectif d'assurer à partir du copyright, le fait que personne ne puisse faire valoir de propriété exclusive sur une oeuvre protégée par cette licence dans le respect de la convention de Berne. La GPL est une mise en oeuvre de la notion de «*copyleft*» [33]. Pour plus d'information voir les travaux officiels du chapitre français de la FSF-Europe [39].

**Logiciel Libre** Les logiciels libres sont des logiciels distribués sous une licence permettant d'utiliser, étudier, modifier, et copier le code librement. [32]

**RFC** «*Request For Comment*» C'est une proposition de norme avec appel à commentaires afin de l'améliorer, la plupart du temps elles deviennent des normes par leur adoption (ex la RFC 822 décrit les en-têtes de courriers électroniques).

**RTFM** vient de l'anglais *Read The F...ing Manual*. Cette injonction dit à une personne qui n'a pas lu les ressources documentaires mises à disposition des utilisateurs. Tout projet Unix dispose d'un manuel électronique installé automatiquement<sup>1</sup>.

**SGI** Silicon Graphic Inc., désormais «SGI», cette société spécialisée dans les produits de conception graphiques de haut de gamme, notamment dans le domaine de la modélisation 3D.

**SUN** Stanford University Network, entreprise cofondée par Bill Joy dans les années 1980. Le but de cette entreprise était de fournir des systèmes Unix sur des plate-formes bon marché de l'époque. Aujourd'hui, cette entreprise est spécialisée dans les infrastructures réseaux centrées autour de leur solution Unix.

**Système d'exploitation** Le système d'exploitation est la partie logicielle la plus proche de la machine. Cette partie a pour but de rendre invisible la nature physique de la machine aux utilisateurs. Elle est utilisée pour lancer des logiciels, gérer les extensions logicielles et matérielles, gérer les fichiers.

---

<sup>1</sup>Originellement, cette réplique est une boutade à l'égard des département des ressources humaines ; au début des systèmes de messageries en entreprise, les boîtes aux lettres étaient inondées de mémos. Ce qui valaient aux membres du personnels étourdis le droit à la remarque RTFMemo. En réponse, Memo devint manuel.