CTS du 07/12/2011 – intervention de l'expert demandé par l'Union syndicale Solidaires Fonction Publique (SUD Travail Affaires Sociales, Solidaires CCRF et SCL, Solidaires IDD) sur les points concernant l'informatique.

Critiques des politiques informatiques menées au ministère du travail

- Vu le schéma directeur
- Vu le projet EUCLIDE
- Vu les choix techniques et d'évolution du SI fait ces dernières années
- Vu le turnover au niveau national.
- Vu le lieu d'emménagement du Cesian

Avertissement : Ce rapport n'est pas une critique exhaustive mais s'appuie et met en avant 6 points extraits du schéma directeur et du projet euclid + un 7ème point sur le lieu d'implantation du cesian qui pose une problématique particulière.

a1 : L'encouragement à unifier les systèmes d'information des différents ministères en omettant de garantir l'usage restreint des bases de données à leurs seuls périmètres initialement autorisés par la CNIL.

Toute décision qui étend le périmètre d'utilisation de données informatisées nominatives en dehors des accords CNIL est illégale, et ce périmètre précise quels sont les fonctionnaires, les services de l'Etat, la région, qui manipulent ces données informatisées, les traitements automatisés autorisés, et pour quelles utilisations....

a2 : L'homogénéisation des infrastructures techniques, sans garantir notre liberté de choix des fournisseurs et des logiciels, et sans encourager l'usage de logiciels open sources.

Qu'assistons-nous ces dernières années :
A 2 choix emblématiques.

1) Le choix de centraliser nationalement les serveurs pour l'application Cap-sitere et de choisir une solution Pivotal basée sur des systèmes d'exploitation Microsoft (Windows Server). Ce choix a été fait malgré la désapprobation des analystes. Il a été fait sous l'emprise de la précipitation. Et les lobbys ont agi en amont.

2) Le choix de la Dagemo de remplacer les serveurs bureautiques Novell de chaque région par des serveurs Windows Microsoft, malgré la désapprobation des analystes. Ce choix Microsoft pour les serveurs a été l'objet d'une débauche de lobbying. Il était pratiquement impossible d'approcher un responsable de la SDSI, d'ouvrir une porte à la Dagemo ou à l'INT lors des formations informatiques, sans tomber sur l'intervention de personne chuchotant aux agents en charge du dossier de choisir la solution Microsoft.

Il est préjudiciable pour l'image des services du Ministère du Travail d'avoir choisie une entreprise en situation de monopole dans le monde telle que Microsoft, pour équiper ses serveurs. (Pour les postes de travail le problème se pose différemment.)

L'autre inconvénient de choisir Microsoft pour les serveurs, sont les nombreux protocoles propriétaires non publiques de Microsoft, mis en oeuvre, faisant que si l'on veut pleinement exploiter les différentes fonctions de ces serveurs il faut obligatoirement choisir des applications Microsoft qui seuls utilisent ces protocoles. D'où perte de liberté dans le choix des fournisseurs et des logiciels. Nous avons donc en quelque sorte les mains liées par la solution tout Microsoft.

Un réseau informatique digne de ce nom est capable d'intégrer toutes sorte d'ordinateurs Windows, Mac, Linux, et d'autres..., du moment qu'ils utilisent les protocoles et standards publics les plus répandus, de communication. Aussi ce n'est pas le matériel qu'il faut homogénéiser mais les protocoles utilisés, et qu'il faut choisir parmi les plus répandus, publics et non propriétaires.

On sépare la question du poste de travail et celle des serveurs. Les enjeux diffèrent. Grosso modo le poste de travail touche directement aux conditions de travail des agents, et le serveur met en oeuvre des traitements sur des données informatiques concernant les usagés, est constituent en cela les coeurs du Système d'Information. Autant le système Windows de Microsoft est adaptée pour le poste de travail pour ses qualités graphiques, autant il ne convient pas pour les serveurs, qui eux n'ont pas besoin d'interface graphique, mais qui ont besoin entre autre d'interopérabilité et de transparence. Les traitements sur les données faits par les serveurs, devraient être faits par des programmes open source, c'est à dire tel que le programme écrit dans un langage évolué soit lisibles par un développeur et soit publique. Ce qui n'est pas le cas avec Microsoft. C'est l'exigence de clarté qu'on peut attendre d'une Fonction publique. La loi semble ne pas prévoir formellement cela. Mais nous pouvons solliciter la réflexion des sénateurs sur cette question.

La Dagemo a choisi une autre voie malgré la réprobation générale des équipes informatiques régionales et sous l'influence des lobbys, qui n'ont pas eu trop de mal du fait de la précarisation de nos responsables de projets. En effet plusieurs de nos chefs de projet ou responsables d'une application au niveau national s'avèrent être des contractuels à duré déterminée, ou des pions déplaçables à volonté, et non des fonctionnaires avec un statut protecteur qui aurait mille fois été nécessaire à ces places. Le nombre de démissions ou de fins de contrats non renouvelés parmi nos référents nationaux atteint un niveau de turnover élevé. Où sont passé nos référents, écartés parce que contrariants des projets obscures en amont, ou parce que leur travail fini, faut tourner la page, et sont-t-il maintenant pour certain embauché par la société à la qu'elle ils ont fait gagner le marché ?.... Concernant le nombre de fonctionnaires ingénieurs informaticiens à la Dagemo, une peau de chagrin, que nous dit le schéma directeur sur cette absence criante de compétences informatiques au plus haut niveau ? Rien ! (Voir sous-traitance peut-être)

Concernant le poste de travail, il est étonnant que l'on choisisse la solution Pack Office 2010 de Microsoft qui est coûteuse, au lieu de la solution Open Office gratuite qui offre des possibilités analogues ou du Pack Office 1997 dont nous sommes toujours propriétaire d'un grand nombres de licences pérennes et qui offre des possibilités analogues. Le coût des licences Pack Offices 2003, puis pour certain Pack Office 2007, puis Pack Office 2010, qu'il faut acquérir est de l'ordre de 300€ par poste...
Alors si l'homogénéisation consiste à financer Microsoft… on n'en veut pas.

a3 : "Professionnalisation, mutualisation et réduction des personnels informatiques."
En d'autre terme, cela signifie externalisation de ce qui peut être externalisé, captage et centralisation des compétences au niveau national, pour que ne reste au niveau local que des tâches dites "presse bouton".

C'est exactement le contraire de ce qu'il faut faire.

Il faut une internalisation, et une réappropriation les traitements informatiques, qu'ils soient faits par des fonctionnaires attitrés, et il faut décentraliser les centres de données et de décisions techniques, pour redonner en région la gestion des bases de données régionales. Réhabiliter les centres de décision informatique au niveau régionale, départemental et sur chaque site, pour que les responsabilités, d'administration, de gestion, et de maintenances, assurées par des fonctionnaires, se rapprochent des agents utilisateurs. Rien ne vaut une explication en tête à tête, avec des personnes responsables.... Comment voir ces responsables s'ils sont à Paris ou on ne sait où ?

Il faut redonner pleinement à l'équipe informatique régionale sa mission de service commun informatique pouvant être sollicité directement par les agents.

Le système d'information s'est concentré nationalement, récupérant un bon nombre de bases de données qui étaient localisées en régions et sous la responsabilité de chaque Direction Régionale, et a externalisé ces tâches de pérennisations des données à un prestataire privé, en l'occurrence British Télécom, faisant que ces données sont manipulées de façon journalière par des employés de droits privés, et concentrées sur deux sites de stockages, un pour la production et un autre pour les sauvegardes.

L'exemple le plus illustratif est sans aucun doute celui de Cap-sitere, répertoriant l'ensemble des enquêtes passées et en cours en droits du travail ainsi que des informations diverses sur les entreprises et leurs représentants du personnels, qui se trouve mémorisé sur des serveurs du Cesian et administré par des employés de droit privé de British Télécom.

Il faut éviter l'écueil d'un centralisme excessif, les bases de données et les centres de décisions techniques doivent être répartis géographiquement. Il faut revenir à une localisation des bases de données en régions. Les données informatisées nominatives en région n'ont pas à priori à être consultée par d'autres régions (voir périmètre de l'accord CNIL). Les fonctionnaires qui manipulent ces données doivent être ceux de la direction régionale concernée. Il faut donc que ces bases soient en région. Et les agents utilisateurs doivent pouvoir demander des comptes directement sur place à des responsables de la région devant avoir réellement les moyens techniques de corriger les dysfonctionnements qui pourraient survenir.

Les traitements faits sur les données dans Cap-sitere, sont fait par des programmes non open source, des programmes propriétaires. Nous ne savons donc pas ce qu'ils font exactement autrement que par les bonnes paroles du prestataire.

Pour les serveurs bureautiques Microsoft installés en région, la Dagemo a fait acte de défiance à l'égard des équipes informatiques régionales en leur retirant la fonction d'administrateur de ces serveurs, seules certaines fonctions d'administration sont conférées aux équipes locales mais pas le compte administrateur des serveurs en région. Cela veut dire que l'on a centralisé les lieux de décision et de mise en oeuvre technique des serveurs régionaux, et cela est, de plus, sous-traité à British Télécom.

Actuellement il existe près de 90 administrateurs nationaux dont la plupart ont un droit d'accès total en tant que membre du groupe Domain Admins, sur les serveurs bureautiques de toutes les régions ainsi qu'a tout les postes de travail branchés sur le réseau et répertorié dans notre annuaire (Active Directory) et qui travaillent étroitement avec British Telecom lorsqu’ils ne sont pas eux même employés de British Telecom ! Pouvez-vous nous garantir que les administrateurs sont bien de nos services ?

Beaucoup trop d'accès hors régions offrant des possibilités de supervision et d'analyse de fichiers, qui misent en oeuvre, par exemple, pour suivre toutes les modifications faites par un agent, seraient

sans aucun doute illégal et constituerait un contrôle abusif du travail et une violation de la vie privé. Mais qu'est ce qu'une telle action illégale s'il n'existe aucun moyen de savoir si une telle action se produit ou pas ? En terme de sécurité à l'égard des indiscrétions, c'est le contraire de ce qu'il faut faire.

Lors de la solution précédente, les serveurs bureautiques étaient sous une architecture Novell, et nous n'avions qu'un seul compte d'administrateur national.

D'autre part côté utilisateurs, dans de nombreux sites, les agents ne sont plus administrateurs de leur poste, ce droit leur étant abusivement retiré, limitant ainsi les possibilités de l'agent de modifier son poste de travail, déplaçant là encore le niveau de responsabilisation, alors que c'est le contraire qu'il faudrait faire, décentraliser et responsabiliser chaque agents afin que celui-ci soit pleinement administrateur de son poste de travail.

A la description de ces frasques nous pourrions peut-être convaincre les sénateurs de mener une commission d'enquête sénatoriale sur les systèmes informatiques et leur dérives depuis quelques années sous l'effet combiné de la RGPP, de la révolution technique, de la centralisation technique, et de l'externalisation hasardeuse, en matière informatique, dans notre ministère.

a4 : Formalisation des principaux processus informatiques et indicateurs de performance associés. Un tableau de bord sera élaboré, qui permettra de suivre périodiquement l'évolution des indicateurs retenus.

Technique ancienne du privé que l'on veut maintenant appliquer à la fonction publique en totale contradiction avec l'esprit du service public, et qui est part ailleurs maintenant abandonnée dans le privé car jugée contreproductive.

Première étape pour pouvoir proposer une externalisation, qui va à l'encontre de ce que nous défendons. Il faut revenir à une situation stricte où la manipulation journalière des données informatisées en vue de leur pérennisation est faite par des fonctionnaires attitrés.

a5 : Réduction de 40 ETP exigé par la mesure RGPP2 509M.

La RGPP est remise en question, voir conclusion du CESE (institution représentative de la vie économique et sociale et environnementale) qui dit stop à la RGPP.
Le Sénat n'est plus favorable à la RGPP.
Les échéances électorales arrivent bientôt pour le renouvellement de l'exécutif.
Est-il encore opportun pour les hauts fonctionnaires de mettre en oeuvre cette mesure ?...

Nous voulons le maintien de tous les postes liés à l'informatique et le redéploiement au plus près des services des informaticiens et des ATI, notamment.
De plus l'impact de ce projet sur les conditions de travail vous impose un examen du projet en CHSCTM avant avis des comités techniques.

a6 : Un besoin local DIRECCTE = une solution unique au plan national.

La formule est trop simple et n'est pas souhaitable. Car il existe des expérimentations en région de solutions à des problèmes locaux. Et lorsqu'une solution s'avère pertinente elle est alors proposée aux autre régions. Voila la richesse qu'apporte la liberté de développement de solution local.

Ce qui manque est le moyen de communiquer cette compétence aux autres régions une fois les premières phases d'expérimentation réussies.

Au lieu de cela, il est proposé de regrouper dans des GAD (Groupe d'Appui au Direccte) les compétences locales à différentes régions pour décider quel projet doit être développé pour toutes les régions. Voir les textes relatifs au développement du logiciel libre dans le monde "la Cathédrale et le Bazar" (voir ci-après en annexe) où il est clairement démontré que l'organisation ainsi hiérarchisée du développement d'un projet n'aboutit pas à de bons résultats, alors que l'organisation juxtaposant librement les expériences de développement d'une multitude de groupes, aboutissent à de meilleurs résultats en apportant des points de vue différents et en permettant l'accès à l'autoformation à un plus grand nombres d'acteurs.

Les agents informaticiens ont besoin de formation continue pour être toujours pleinement acteurs de la révolution technologique informatique que nous vivons tous. Cette formation comprend la veille technologique qui ne saurait être réservé à une élite au sein des GAD.

a7 : Le déménagement du Cesian dans les locaux de la Direction Générale de la Gendarmerie Nationale à Rosny-sous-Bois.

Le Cesian qui mène les activités de support et de production de l'infrastructure du Ministère, qui est grosso modo la plate-forme support des EIR (Équipes Informatiques Régionales), et également des équipes nationales, et qui regroupe de nombreux serveurs d'applications nationales dont Cap-sitere, dans notre ministère, déménage à la DGGN de Rosny-sous-Bois. Sa situation géographique n'est pas une question anodine, et son installation qui est actuellement en cours dans les locaux de la DGGN (Direction Générale de la Gendarmerie Nationale), transgresse le principe d'indépendance de l'inspection du travail, et d'une façon plus générale, ne respecte pas la séparation des services.

En effet, les données du ministère du travail ne sont pas celles de la gendarmerie nationale. Et comment établir dans les faits cette distinction, si nos serveurs de productions qui hébergent nos données informatisées sont dans les mêmes locaux ?

Les services des autres ministères, et plus particulièrement la gendarmerie et la police, d'autant plus qu'ils représentent des pouvoirs coercitifs importants, n'ont pas accès aux données nominatives informatisées du ministère du Travail. Exception faite, exemple illustratif, dans le cas d'une commission rogatoire ordonnée par un juge d'instruction, un juge indépendant de l'exécutif, saisi d'une affaire criminelle, qui a pour mission de faire « tout acte utile à la manifestation de la vérité », et qui mène une instruction à charge et à décharge, dans le respect des libertés de la défense des parties mises en cause et des principes qui régissent la justice en France.

Si les serveurs sont dans les mêmes locaux que les gendarmes, la commission rogatoire n'est alors plus nécessaire, et la loi est abusivement contournée. Il faut donc nécessairement un bail privatif, faisant que ces locaux ne sont plus ceux de la DGGN mais loués au Ministère du Travail. Et malgré cela, une telle situation peut porter préjudice en termes de manque de confidentialité des données du Ministères du Travail vis-à-vis des services de la gendarmerie. Car il peut être trop facile d'accéder sur place, d'une manière ou d'une autre, à ces informations, sans pour autant commettre une effraction manifeste. La séparation n'est pas assez franche. Il faut une séparation nette de telle sorte que toute fuite de données nécessite nécessairement une effraction manifeste, une infraction caractérisée. C'est un principe de droit en matière de sécurité pour dissuader un usage abusif.

Annexe

Extraits de « La cathédral et le bazar »

- - - - - - - -

Auteur : Eric S. Raymond (esr@thyrsus.com)
Traducteur : Sébastien Blondeel

Date: 1998/08/11 20:27:29

J'analyse le succès d'un projet de logiciel dont le code source est ouvert,
NdT traduction de open-source software, terme par lequel l'auteur a délibérément remplacé le terme précédent free software (logiciel libre). ``Ouvert'' signifie ici à la fois public et susceptible d'être commenté et corrigé par ceux qui s'y intéresseront suffisamment.
fetchmail (``va chercher le courrier''), qui a été lancé délibérément pour tester certaines théories surprenantes du génie logiciel suggérées par l'histoire de Linux. Je discute ces théories en termes de deux styles de développement fondamentalement différents, le modèle ``cathédrale'' de la majorité du monde commercial, à opposer au modèle ``bazar'' du monde de Linux. Je montre que ces modèles dérivent d'hypothèses opposées sur la nature de la tâche consistant à déboguer un logiciel. Enfin, je m'efforce de démontrer, à partir de l'expérience apportée par Linux, qu'``Étant donnés suffisamment d'observateurs, tous les bogues sautent aux yeux'', je suggère des analogies productives avec d'autres systèmes auto-correcteurs par agents égoïstes, et je conclus en réfléchissant aux implications de ces idées sur l'avenir du logiciel.

---

1. La cathédrale et le bazar

Linux est subversif. Qui aurait imaginé, il y a seulement cinq ans, qu'un système d'exploitation de classe internationale prendrait forme comme par magie à partir de bidouilles

NdT Aux termes *to hack, hacker*, Eric S. Raymond a consacré un article complet. Il s'agit de l'action de comprendre comment les choses fonctionnent, de vouloir savoir ce qui se cache dans les mécaniques diverses, et de les réparer ou de les améliorer. En aucun cas, ce terme ne doit être confondu, sous cette plume, avec les pirates de l'informatique, ou *crackers*. J'ai choisi de traduire ce terme par ``bidouille'', et je vous demande de ne pas y attacher d'a priori négatif. Un bidouilleur construit des choses parfois très belles et très compliquées, alors qu'un pirate cherche à détruire.
faites pendant le temps libre de plusieurs milliers de développeurs disséminés de par le monde, et reliés seulement par les les liens ténus de l'Internet ?

Certainement pas moi. Quand, début 1993, Linux est apparu pour la première fois sur mon écran radar, cela faisait déjà dix ans que j'étais impliqué dans le développement sous Unix et dans la programmation de logiciels dont le code source est ouvert. Au milieu des années 1980, j'étais l'un des premiers contributeurs à GNU. J'avais distribué sur le réseau une bonne quantité de logiciels dont le code source est ouvert (nethack, les modes VC et GUD pour Emacs, xlife, et d'autres), encore largement utilisés de nos jours. Je pensais savoir comment tout cela fonctionnait.

Linux a remis en cause une grande partie de ce que je croyais savoir. J'avais prêché l'évangile selon Unix sur l'utilisation de petits outils, le prototypage rapide et la programmation évolutive, depuis des années. Mais je pensais aussi qu'il existait une certaine complexité critique au delà de laquelle une approche plus centralisée, plus a priori, était nécessaire. Je pensais que les logiciels les plus importants (comme les systèmes d'exploitation et les très gros outils comme Emacs) devaient être conçus comme des cathédrales, soigneusement élaborés par des sorciers isolés ou des petits groupes de mages travaillant à l'écart du monde, sans qu'aucune version bêta ne voie le jour avant que son heure ne soit venue.

Le style de développement de Linus Torvalds - distribuez vite et souvent, déléguez tout ce que vous pouvez déléguer, soyez ouvert jusqu'à la promiscuité - est venu comme une surprise. À l'opposé de la construction de cathédrales, silencieuse et pleine de vénération, la communauté Linux paraissait plutôt ressembler à un bazar, grouillant de rituels et d'approches différentes (très justement symbolisé par les sites d'archives de Linux, qui acceptaient des contributions de *n'importe qui*) à partir duquel un système stable et cohérent ne pourrait apparemment émerger que par une succession de miracles.

Le fait que ce style du bazar semblait fonctionner, et bien fonctionner, fut un choc supplémentaire. Alors que j'apprenais à m'y retrouver, je travaillais dur, non seulement sur des projets particuliers, mais encore à essayer de comprendre pourquoi le monde Linux, au lieu de se disloquer dans la confusion la plus totale, paraissait au contraire avancer à pas de géant, à une vitesse inimaginable pour les bâtisseurs de cathédrales.

Vers le milieu de 1996, je pensais commencer à comprendre. Le hasard m'a donné un moyen incomparable de mettre ma théorie à l'épreuve, sous la forme d'un projet dont le code source est ouvert et que je pourrais consciemment faire évoluer dans le style du bazar. Ce que je fis -- et je rencontrai un franc succès.

Dans le reste de cet article, je conterai l'histoire de ce projet, et je l'utiliserai pour proposer des aphorismes relatifs au développement efficace de logiciels dont le code source est public. Je n'ai pas appris toutes ces choses dans le monde Linux, mais nous verrons de quelle manière le monde Linux les place au devant de la scène. Si je ne me trompe pas, elles vous aideront à comprendre exactement ce qui fait de la communauté Linux une telle manne de bons logiciels - et à devenir plus productif vous-même.

- - - - - - - -

…/…

- - - - - - - -

9. Prérequis nécessaires au style bazar

Les premiers relecteurs et les premiers publics auprès desquels cet article a été testé ont posé de manière régulière la question des prérequis nécessaires à la réussite d'un développement dans le style bazar, en incluant dans cette question aussi bien les qualifications du chef de projet que l'état du code au moment où il est rendu public et tente de rallier autour de lui une communauté de co-développeurs.

Il est assez évident qu'il n'est pas possible de commencer à coder dans le style bazar dès le début. On peut tester, déboguer et améliorer un programme dans le style bazar, mais il sera très difficile de *faire démarrer* un projet dans le mode bazar. Linus ne l'a pas tenté, moi non plus. Il faut que votre communauté naissante de développeurs ait quelque chose qui tourne, qu'on puisse tester, avec quoi elle puisse jouer.

Quand vous initiez un travail de développement en communauté, il vous faut être capable de présenter une *promesse plausible*. Votre programme ne doit pas nécessairement fonctionner très bien. Il peut être grossier, bogué, incomplet, et mal documenté. Mais il ne doit pas manquer de convaincre des co-développeurs potentiels qu'il peut évoluer en quelque chose de vraiment bien dans un futur pas trop lointain.

Linux et fetchmail furent tous deux rendus publics avec des conceptions simples, fortes et séduisantes. Nombreux sont ceux qui, après avoir réfléchi au modèle du bazar tel que je l'ai présenté ici, ont très correctement considéré que cela était critique, et en ont rapidement conclu qu'il était indispensable que le chef de projet fasse preuve au plus haut point d'astuce et d'intuition dans la conception.

Mais Linus se fonda sur la conception d'Unix. Ma conception provenait initialement du vieux programme popclient (bien que beaucoup de choses aient changé à terme, proportionnellement bien plus que dans Linux). Alors faut-il que le chef/coordinateur d'un effort commun mené dans le style bazar ait un talent exceptionnel pour la conception, ou peut-il se contenter d'exalter le talent des autres ?

Je pense qu'il n'est pas critique que le coordinateur soit capable de produire des conceptions exceptionnellement brillantes. En revanche, il est absolument critique que le coordinateur soit capable de *reconnaître les bonnes idées de conception des autres*.

Les projets Linux et fetchmail en sont tous deux la preuve. Linus, bien qu'il ne soit pas (comme nous l'avons vu précédemment) un concepteur spectaculairement original, a démontré une puissante capacité à reconnaître une bonne conception et à l'intégrer au noyau Linux. Et j'ai déjà décrit comment l'idée la plus puissante dans la conception de fetchmail (faire suivre le courrier vers le port SMTP) me fut soufflée par quelqu'un d'autre.

Les premières personnes auxquelles j'ai présenté cet article m'ont complimenté en me suggérant que je suis prompt à sous-évaluer dans la conception des projets menés dans le style bazar, leur originalité -- principalement parce que j'en suis pas mal pourvu moi-même, ce qui me conduit à considérer cela comme acquis. Il peut y avoir un fond de vérité là dedans; la conception (à l'opposé de la programmation ou du débogage) est certainement mon point fort.

Mais le problème avec l'intelligence et l'originalité dans la conception de logiciels, c'est que ça devient une habitude -- presque par réflexe, vous commencez à faire dans l'esthétique et le

compliqué alors qu'il faudrait rester simple et robuste. Certains de mes projets n'ont jamais abouti à cause de cette erreur, mais ce ne fut pas le cas pour fetchmail.

Aussi suis-je convaincu que le projet fetchmail a réussi en partie parce que j'ai refoulé ma tendance à donner dans la subtilité; cela contredit (au moins) l'idée que l'originalité dans la conception est essentielle dans des projets réussis menés dans le style bazar. Et pensez à Linux. Imaginez que Linus ait tenté de mettre en pratique des innovations fondamentales dans la conception de systèmes d'exploitation au cours du développement; est-il probable que le noyau qui en résulterait soit aussi stable et populaire que celui que nous connaissons ?

Il faut, bien sûr, une certaine habileté dans la conception et le codage, mais je pense que quiconque envisage sérieusement de lancer un projet dans le style en possède déjà le minimum nécessaire. Les lois du marché de la réputation interne au monde du logiciel dont le code source est ouvert exercent des pressions subtiles décourageant ceux qui pensent mettre à contribution d'autres développeurs alors qu'ils n'ont pas eux-mêmes la compétence d'assurer le suivi. Jusqu'à présent tout cela semble avoir très bien marché.

Il existe une autre qualité qui n'est pas normalement associée au développement logiciel, mais je pense qu'elle est aussi importante qu'une bonne intelligence de la conception pour les projets de style bazar -- et elle est peut-être bien plus importante. Le chef, ou coordinateur, d'un projet dans le style bazar doit être bon en relations humaines et avoir un bon contact.

Cela devrait être évident. Pour construire une communauté de développement, il vous faut séduire les gens, les intéresser à ce que vous faites, et les encourager pour les petits bout du travail qu'ils réalisent. De bonnes compétences techniques sont essentielles, mais elles sont loin de suffire. La personnalité que vous projetez compte aussi.

Cela n'est pas une coïncidence que Linus soit un chic type qu'on apprécie volontiers et qu'on a envie d'aider. Cela n'est pas fortuit non plus que je sois un extraverti énergique qui aime animer les foules et qui se comporte et réagit comme un comique de théâtre. Pour que le modèle du bazar fonctionne, une petite touche de charme et de charisme aide énormément.

10. Le contexte social du logiciel dont le code source est ouvert

Cela est bien connu: les meilleures bidouilles commencent en tant que solutions personnelles aux problèmes de tous les jours rencontrés par leur auteur, et elles se répandent parce que ce problème est commun à de nombreuses personnes. Cela nous ramène à la règle numéro 1, reformulée de manière peut-être plus utile:

18. Pour résoudre un problème intéressant, commencez par trouver un problème qui vous intéresse.

Ainsi fut-il de Carl Harris et du vieux popclient, ainsi fut-il de moi et de fetchmail. Mais cela est bien compris depuis longtemps. Ce qui compte, le détail sur lequel les histoires de Linux et de fetchmail semblent nous demander de nous concentrer, est l'étape suivante -- l'évolution du logiciel en présence d'une communauté d'utilisateurs et de co-développeurs importante et active.

Dans ``Le mythe du mois-homme'', Fred Brooks observa qu'on ne peut pas diviser et répartir le temps du programmeur; si un projet a du retard, lui ajouter des développeurs ne fera qu'accroître son retard. Il explique que les coûts de communication et de complexité d'un projet augmentent de manière quadratique avec le nombre de développeurs, alors que le travail réalisé n'augmente que

linéairement. Depuis, cela est connu sous le nom de ``loi de Brooks", et on la considère en général comme un truisme. Mais si seule la loi de Brooks comptait, Linux serait impossible.

Le classique de Gerald Weinberg ``La psychologie de la programmation sur ordinateur" apporta ce qu'on pourrait considérer après coup comme une correction vitale à Brooks. Dans sa discussion sur la ``programmation non égoïste", Weinberg observa que dans les boîtes où les programmeurs ne marquent pas le territoire de leur code, et encouragent les autres à y chercher les bogues et les améliorations potentielles, les améliorations sont drastiquement plus rapides qu'ailleurs.

Les termes choisis par Weinberg l'ont peut-être empêché de recevoir toute la considération qu'il méritait -- et l'idée que les bidouilleurs sur Internet puissent être décrits comme ``non égoïstes" fait sourire. Mais je pense que cet argument semble aujourd'hui plus vrai que jamais.

L'histoire d'Unix aurait dû nous préparer à ce que nous découvrons avec Linux (et à ce que j'ai expérimenté à une échelle plus modeste en copiant délibérément les méthodes de Linus): alors que l'acte de programmation est essentiellement solitaire, les plus grandes bidouilles proviennent de la mise à contribution de l'attention et de la puissance de réflexion de communautés entières. Le développeur qui n'utilise que son propre cerveau dans un projet fermé ne tiendra pas la route face au développeur qui sait comment créer un contexte ouvert, susceptible d'évoluer, dans lequel la traque des bogues et les améliorations sont effectuées par des centaines de gens.

Mais le monde traditionnel d'Unix n'a pas poussé cette approche dans ses derniers retranchements pour plusieurs raisons. L'un d'entre eux concernait les contraintes légales des diverses licences, secrets de fabrication, et autres intérêts commerciaux. Un autre (mieux compris plus tard) était que l'Internet n'était pas encore assez mûr.

Avant que les coûts d'accès à Internet ne chutent, il existait quelques communautés géographiquement compactes dont la culture encourageait la programmation ``non égoïste" à la Weinberg, et un développeur pouvait facilement attirer tout un tas de co-développeurs et autres touche-à-tout doués. Les laboratoires Bell, le laboratoire d'intelligence artificielle de l'institut de technologie du Massachusetts (MIT), l'université de Californie à Berkeley, devinrent les foyers d'innovations légendaires qui sont restés puissants.

Linux fut le premier projet qui fit un effort conscient et abouti pour utiliser le *monde entier* comme réservoir de talent. Je ne pense pas que cela soit une coïncidence que la période de gestation de Linux ait coïncidé avec la naissance du World Wide Web, ni que Linux ait quitté le stade de la petite enfance en 1993-1994, au moment où l'intérêt général accordé à Internet et que l'industrie des fournisseurs d'accès explosèrent. Linus fut le premier à comprendre comment jouer selon les nouvelles règles qu'un Internet omniprésent rendait possibles.

Même s'il fallait qu'Internet ne coûte pas cher pour que le modèle Linux se développe, je ne pense pas que cela soit suffisant. Il y avait un autre facteur vital: le développement d'un style de direction de projet et d'un ensemble de coutumes de coopération qui permettaient aux développeurs d'attirer des co-développeurs et de rentabiliser au maximum ce nouveau média.

Quel est donc ce style de direction, quelles sont donc ces coutumes ? Ils ne peuvent pas être fondés sur des rapports de force -- même si c'était le cas, la direction par coercition ne produirait pas les résultats qu'on peut observer. Weinberg cite fort à propos l'autobiographie de Pyotr Alexeyvitch Kropotkine, anarchiste russe du XIXe siècle, ``Mémoires d'un révolutionnaire":

``Élevé dans une famille possédant des serfs, j'entrai dans la vie active, comme tous les jeunes gens de mon époque, avec une confiance aveugle dans la nécessité de commander, d'ordonner, de brimer,

de punir et ainsi de suite. Mais quand, assez tôt, je dus diriger d'importantes affaires et côtoyer des hommes *libres*, et quand chaque erreur pouvait être immédiatement lourde de conséquences, je commençai à apprécier la différence entre agir selon les principes du commandement et de la discipline et agir selon le principe de la bonne intelligence. Le premier fonctionne admirablement dans un défilé militaire, mais ne vaut rien dans la vie courante, où on ne peut atteindre son but que grâce à l'effort soutenu de nombreuses volontés travaillant dans le même sens."

``L'effort soutenu de nombreuses volontés travaillant dans le même sens", c'est exactement ce qu'un projet comme Linux demande -- et le ``principe de commandement" est en effet impossible à appliquer aux volontaires de ce paradis de l'anarchie que nous appelons Internet. Pour opérer et se mesurer les uns aux autres de manière efficace, les bidouilleurs qui cherchent à prendre la tête d'un projet de collaboration commune doivent apprendre à recruter et à insuffler de l'énergie dans les communautés d'intérêts convergents à la manière vaguement suggérée par le ``principe de bonne intelligence" de Kropotkine. Ils doivent apprendre à utiliser la loi de Linus.

Plus haut, je me référais à ``l'effet de Delphes" comme une possible explication de la loi de Linus. Mais on ne peut s'empêcher de penser à des analogies très puissantes avec des systèmes adaptatifs en biologie et en économie. Le monde Linux sous de nombreux aspects, se comporte comme un marché libre ou un écosystème, un ensemble d'agents égoïstes qui tentent de maximiser une utilité, ce qui au passage produit un ordre spontané, auto-correcteur, plus élaboré et plus efficace que toute planification centralisée n'aurait pu l'être. C'est ici qu'il faut chercher le ``principe de bonne intelligence".

La ``fonction d'utilité" que les bidouilleurs Linux maximisent n'est pas classiquement économique, c'est l'intangible de leur propre satisfaction personnelle et leur réputation au sein des autres bidouilleurs. (On peut être tenté de penser que leur motivation est ``altruiste", mais c'est compter sans le fait que l'altruisme est en soi une forme d'égoïsme pour l'altruiste). Les cultures volontaires qui fonctionnent ainsi ne sont pas si rares; j'ai déjà participé à des projets semblables dans les fanzines de science-fiction, qui au contraire du monde de la bidouille, reconnaissent explicitement que l'``egoboo" (le fait que sa réputation s'accroisse auprès des autres fans) est le principal moteur qui propulse les activités volontaires.

Linus, en se positionnant avec succès comme gardien des clés d'un projet où le développement est surtout fait par les autres, et en y injectant de l'intérêt jusqu'à ce qu'il devienne auto-suffisant a montré une intelligence profonde du ``principe de bonne intelligence" de Kropotkine. Cette vision quasi-économique du monde de Linux nous permet de comprendre comment cette bonne intelligence est mise en oeuvre.

On peut analyser la méthode de Linus comme un moyen de créer de manière efficace, un marché de l'``egoboo" -- relier les égoïsmes individuels des bidouilleurs aussi fermement que possible, dans le but de réaliser des tâches impossibles sans une coopération soutenue. Avec le projet fetchmail, j'ai montré (à une échelle plus modeste, je vous l'accorde) qu'on peut dupliquer ses méthodes avec de bons résultats. Peut-être même l'ai-je fait un peu plus consciemment et plus systématiquement que lui.

Nombreux sont ceux (en particulier ceux dont les opinions politiques les font se méfier des marchés libres) qui s'attendraient à ce que la culture d'égoïstes sans autre maître qu'eux mêmes soit fragmentée, territoriale, source de gâchis, pleine de petits secrets, et hostile. Mais cette idée est clairement mise en défaut par (pour ne donner qu'un seul exemple) l'époustouflante variété, qualité et profondeur de la documentation relative à Linux. Il est pourtant légendaire que les programmeurs *détestent* écrire des documentations; comment se fait-il, alors, que les bidouilleurs de Linux en produisent tant ? Il est évident que le marché libre d'egoboo de Linux fonctionne mieux que tout

autre pour générer des comportements vertueux, serviables, mieux que les services documentaires largement subventionnés des producteurs de logiciels commerciaux.

Les projets fetchmail et du noyau de Linux montrent tous deux qu'en flattant à bon escient l'ego de beaucoup d'autres bidouilleurs, un coordinateur-développeur fort peut utiliser l'Internet pour tirer parti du fait d'avoir énormément de co-développeurs sans que le projet ne s'effondre en un capharnaüm chaotique. C'est pourquoi je propose la contrepartie suivante à la loi de Brooks:

19: Pour peu que le coordinateur du développement dispose d'un moyen de communication au moins aussi bon que l'Internet, et pour peu qu'il sache comment mener ses troupes sans coercition, il est inévitable qu'il y ait plus de choses dans plusieurs têtes que dans une seule.

Je pense qu'à l'avenir, le logiciel dont le code source est ouvert sera de plus en plus entre les mains de gens qui savent jouer au jeu de Linus, des gens qui abandonnent les cathédrales pour se consacrer entièrement au bazar. Cela ne veut pas dire que les coups de génie individuels ne compteront plus; je pense plutôt que l'état de l'art du logiciel dont le code source est ouvert appartiendra à ceux qui commencent par un projet individuel génial, et qui l'amplifieront à travers la construction efficace de communautés d'intérêt volontaires.

Et peut-être même qu'il n'est pas question ici que de l'avenir du logiciel *dont le code source est ouvert*. Aucun développeur qui fonctionne avec un code source fermé ne peut mobiliser autant de talent que celui que la communauté Linux peut consacrer à un problème donné. Bien rares même sont ceux qui auraient eu les moyens de rémunérer les deux cents et quelques contributeurs à fetchmail !

Peut-être qu'à terme, la culture du logiciel dont le code source est ouvert triomphera, non pas parce qu'il est moralement bon de coopérer, non pas parce qu'il est moralement mal de ``clôturer'' le logiciel (en supposant que vous soyez d'accord avec la deuxième assertion; ni Linus ni moi ne le sommes), mais simplement parce que le monde dont le code source est fermé ne peut pas gagner une course aux armements évolutive contre des communautés de logiciel libre, qui peuvent mettre sur le problème un temps humain cumulé plus important de plusieurs ordres de grandeurs.

11. Remerciements

Un grand nombre de personnes m'ont aidé, au travers de conversations, à améliorer cet article. Je remercie tout particulièrement Jeff Dutky <dutky@wam.umd.edu> qui m'a suggéré la formule ``le débogage est parallélisable'', et qui m'a aidé à tirer l'analyse qui en découle. Merci aussi à Nancy Lebovitz <nancyl@universe.digex.net> qui m'a suggéré que je suivais l'exemple de Weinberg en citant Kropotkine. J'ai reçu aussi des critiques utiles de la part de Joan Eslinger <wombat@kilimanjaro.engr.sgi.com> et de Marty Franz <marty@net-link.net> de la liste de General Technics. Je remercie les membres du PLUG, le groupe d'utilisateurs de Linux de Philadelphie (jeu de mot signifiant aussi ``branché''), pour avoir joué le rôle du premier public critique de cet article. Enfin, les commentaires de Linus Torvalds me furent utiles, et son soutien des premières heures m'encouragea beaucoup.

12. Pour aller plus loin

J'ai cité plusieurs extraits du classique de P. Brooks *The Mythical Man-Month* (*Le mythe du mois-homme*) parce que, sous bien des aspects, ses conclusions doivent encore être affinées. Je recommande chaleureusement l'édition faite par la société Addison-Wesley (ISBN 0-201-83595-9)

à l'occasion du vingt-cinquième anniversaire de ce livre, qui ajoute son article de 1986 intitulé ``*No Silver Bullet''* (pas de balle en argent).

Cette nouvelle édition est préfacée par une inestimable rétrospective des 20 dernières années dans laquelle Brooks reconnaît les quelques affirmations de son texte original qui n'ont pas résisté à l'épreuve du temps.

J'ai lu la rétrospective pour la première fois après avoir écrit la quasi totalité du présent article, et j'ai été surpris de constater que Brooks attribue des pratiques de type bazar à Microsoft !

De Gerald M. Weinberg, *The Psychology Of Computer Programming* (la psychologie de la programmation des ordinateurs) (New York, Van Nostrand Reinhold 1971) introduisit le concept assez maladroitement appelé de ``programmation non égoïste''. Bien que n'étant pas le premier à se rendre compte de la futilité du ``principe de commandement'', il fut probablement le premier à reconnaître cet aspect et à argumenter sur ce dernier par rapport au développement logiciel.

Richard P. Gabriel, en contemplant la culture Unix de l'avant-Linux, argumenta à contrecoeur en faveur de la supériorité d'un modèle de type bazar primitif dans son article de 1989 *Lisp: Good News, Bad News, and How To Win Big* (Lisp: bonnes nouvelles, mauvaises nouvelles, et comment gagner gros). Bien qu'il ait mal vieilli sous certains aspects, cet essai est toujours adulé à juste titre dans les cercles de fans de Lisp (dont je fais partie). Un correspondant m'a rappelé que la section intitulée ``Worse Is Better'' (le pire est l'ami du mieux) peut presque se lire comme une anticipation de Linux. On peut trouver ce papier sur la Toile (WWW) à l'adresse http://www.naggum.no/worse-is-better.html.

Peopleware: Productive Projects and Teams (ressources humaines: projets et équipes productifs) de De Marco et Lister's (New York; Dorset House, 1987; ISBN 0-932633-05-6) est un joyau méconnu, et j'ai été enchanté de voir Fred Brooks le citer dans sa rétrospective. Bien que la matière directement applicable au monde Linux ou au monde du logiciel dont le code source est ouvert en général soit rare, les auteurs ont de bonnes intuitions sur les prérequis nécessaires au travail créatif qui intéresseront quiconque envisage d'importer les vertus du modèle du bazar dans un contexte commercial.

Enfin, je dois admettre que j'ai failli appeler cet article ``La cathédrale et l'agora'', du mot grec désignant un marché ouvert ou un lieu public de rencontres. Les articles fondateurs ``agoric systems'' (systèmes agoriques), de Mark Miller et Eric Drexler, en décrivant les propriétés émergentes d'écosystèmes informatiques similaires au libre marché, m'ont aidé à avoir les idées plus claires sur des phénomènes analogues dans la culture du logiciel dont le code source est ouvert quand Linux m'a mis le nez dedans, cinq ans plus tard. On peut trouver ces papiers sur la Toile à l'adresse http://www.agorics.com/agorpapers.html.

- - - - - - - -

…/…

- - - - - - - -