Migration vers les logiciels libres

Mohamadi ZONGO
mzongo@zcp.bf

Kassim ASSIROU
assirou@gmail.com

Atelier Migration
Rabat -RALL 2007

ubuntu

Enjeux et organisation de la migration

ubuntu

Enjeux et organisation

- Pourquoi migrer ?
- Préalables
- Au niveau serveur
- Au niveau poste de travail
- Méthodologie

Pourquoi migrer ?

Aujourd'hui au sein de plusieurs organisations, les ordinateurs utilisant les logiciels libres sont capables de fournir une alternative entièrement fonctionnelle et rentable aux logiciels propriétaires.

La décision de migrer ou ne pas migrer votre environnement dépend de plusieurs facteurs.

Il est important de considérer soigneusement tous les défis techniques et organisationnels avant de prendre toute décision.

Pourquoi migrer ?

Il est parfois très difficile de bâtir une stratégie d'intégration des logiciels libres en se basant uniquement sur des performances techniques.

Mais dans une analyse plus large, une migration vers ces environnements pourrait ne pas être néanmoins justifiable en raison du coût élevé à surmonter par l'organisation déjà enfermée dans un environnement propriétaire.

Pourquoi migrer ?

Certains arguments contribuent à différencier les logiciels libres et sont stratégiques pour justifier la migration :

- la sécurité du système
- les coûts directs et indirects
- l'administration du système
- la nécessité d'utiliser des standard ouverts
- l'adaptabilité des environnements
- les principes et la philosophie des logiciels libres
- l'élimination des évolutions imposées
- les économies d'échelle

Préalable

- Avant de commencer, avoir une compréhension claire des motifs de la migration
- S'assurer qu'il existe un support actif du changement de l'équipe informatique et des utilisateurs
- S'assurer qu'il existe un décideur, volontaire en faveur de la migration, dont la position hiérarchique soit la plus élevée possible
- Bâtir une expertise et un réseau de relations avec la communauté logiciel libre
- Commencer avec des systèmes non critiques
- S'assurer que chaque étape de la migration soit gérable

Au niveau des Serveurs

Pour les serveurs, l'opportunité du logiciel libre est déjà bien acquise et largement déployée.

La migration de serveurs peut généralement être réalisée sans effet de bord sur les utilisateurs. C'est en principe un bon point de départ.

Au niveau des postes de travail

Le déploiement du logiciel libre à l'échelle du poste de travail présente les meilleures opportunités d'économies.

Lors de la migration de ceux-ci, les nouvelles applications libres devront interopérer avec les applications existantes.

En particulier, l'aspect interopérabilité du travail de groupe entre les systèmes libres et les systèmes propriétaires doit être étudié.

Au niveau des postes de travail

Lors du remplacement de logiciels propriétaires de bureautique, divers prototypes devront être testés pour s'assurer qu'ils produisent des résultats corrects.

Les macros doivent être ré-écrites - de préférence sous forme de scripts. Les applications pour lesquelles aucun équivalent libre n'existe peuvent tourner sur des clients légers.

Au cours du temps, les applications de poste de travail peuvent être remplacées progressivement par leurs équivalents open source.

Méthodologie

Tout projet de migration consiste généralement en :

- une phase de collecte d'informations et de définition de projet
- une justification de la migration incluant le calcul des coûts associés
- une ou plusieurs phases pilotes conçues pour tester le plan et la justification. Les données issues de ces pilotes peuvent ensuite être réinjectées dans le modèle de coûts
- le déroulement du plan
- le suivi et l'actualisation de l'expérience réelle acquise par rapport au plan

Méthodologie

La définition du projet

- **La Description de l'ensemble des conditions initiales** :
  - architecture(s) système
  - applications et leurs données associées
  - protocoles et standards utilisés
  - matériel et environnement physique
  - pré-requis sociaux tels que les langues et ensembles de particularités des équipes

- **L'ensemble de conditions cibles du même niveau de détail**

- **Description de la démarche permettant le passage des**

ubuntu

Processus de migration

ubuntu

Processus de Migration

1. Mise en place d'une équipe projet
2. Comprendre l'environnement cible
3. Redéfinir l'architecture si possible
4. Maîtriser le concept logiciel libre
5. Analyse des systèmes existants
6. Evaluer un cas concret et détaillé de migration
7. Consulter les utilisateurs
8. Démarrer un projet pilote
9. Définir la vitesse de migration
10. Réaliser la migration de l'ensemble de l'organisation
11. Faire un suivi des retours utilisateurs et résoudre les problèmes

Processus de Migration

- 1. Mise en place d'une équipe

1. Mise en place d'une équipe

Monter une équipe avec les compétences nécessaires, un **Chef de Projet** et un appui managérial. Cet appui est important pour éviter les résistances au changement depuis la norme des systèmes propriétaires.

L'appui devra être suffisant pour permettre au moins la mise en place de pilotes représentatifs, ainsi un cas concret de base sera-t-il sans doute nécessaire.

Processus de Migration

- 2. Comprendre l'environnement cible

2. Comprendre l'environnement cible

Comprendre l'environnement cible, aussi bien le logiciel libre que l'architecture de base, ainsi que les options et choix possibles.

Cela implique la formation de l'équipe existante, du recrutement ou le recours à des consultants. Cela entraîne un certain coût initial et nécessite donc l'appui managérial suffisant. Parfois l'on s'attend à ce qu'un logiciel gratuit puisse être compris et utilisé à coût zéro. Ce n'est pas le cas.

3. Redéfinir l'architecture si possible

3. Redéfinir l'architecture si possible

La migration est une opportunité pour re-visiter l'architecture de base aussi bien que les logiciels applicatifs.

L'architecture préconisée s'appuie sur un contrôle centralisé (authentification, autorisations, administration).

Il peut y avoir des implications financières de ces changements qui doivent être prises en compte.

4. Maîtriser le concept logiciel libre

4. Maîtriser le concept logiciel libre

- **Comprendre les licences** : les implications des licences logiciel libre doivent être clairement perçues en particulier si l'organisation peut être amenée à distribuer des modifications aux logiciels. Se reporter aux documents indiqués dans l'introduction pour les détails.
- **Étudier comparativement des logiciels équivalents** : lorsqu'il existe plusieurs choix pour une seule fonction - il existe par exemple au moins trois tableurs de qualité en logiciel libre - les administrateurs doivent peser le pour et le contre de chaque produit.

4. Maîtriser le concept logiciel libre

- **Étudier les différences entre les distributions** : certaines sont appuyées par des entreprises commerciales qui fournissent un support et des mises à jour. D'autres ont des caractéristiques spécifiques à leur avantage. Toutes ces différences doivent être étudiées avant qu'un choix soit effectué.

4. Maîtriser le concept logiciel libre

- **Déterminer le niveau de support nécessaire** : un support payant peut être obtenu dans certains cas auprès des développeurs de l'application ou de la distribution. Dans le cas contraire, des entreprises tierces peuvent fournir un support en raison de la disponibilité du code source et de nombreuses entreprises proposent un tel support.

Il y a une différence marquante par rapport au marché du logiciel propriétaire où un support détaillé ne peut être fourni que par les entreprises qui ont le privilège d'avoir accès aux sources. Cela devient dangereux si l'éditeur

4. Maîtriser le concept logiciel libre

- **Déterminer le niveau de support nécessaire** : si tout le reste échoue, de nombreuses applications disposent de listes de diffusion actives sur lesquelles les questions et demandes d'assistances reçoivent des réponses de personnes qui s'intéressent à celles-ci. La présence d'une liste de diffusion active et d'une communauté d'utilisateurs est souvent l'un des critères initiaux de sélection de composants logiciels.

5. Analyse des systèmes existants

Cette étape ne sera pas utile seulement pour réaliser la migration proprement dite mais elle sera aussi nécessaire pour bâtir un modèle de coût total de possession (TCO - Total cost of ownership) pour un cas concret détaillé.

5. Analyse des systèmes existants

1- Détail de chaque application utilisée

- nom de l'application, numéro de version et point de contact
- nombre d'utilisateurs
- système d'exploitation, liste des systèmes d'exploitations possibles y compris les environnements de type Citrix
- interdépendance avec d'autres applications aussi bien sur le client que sur le serveur
- matériel nécessaire, en particulier matériels non standard ou spécifiques
- protocoles de communications utilisés
- formats de fichiers nécessaires
- internationalisation et localisation nécessaires (nécessité de

5. Analyse des systèmes existants

2- Les contraintes sur les données

Il s'agit des données textes et tableaux, données sonores/vocales et visuelles ainsi que les bases de données :

- contraintes d'interfaçage avec des systèmes ou utilisateurs hors du contrôle de l'organisation

- contraintes d'exploitabilité future des données, existence d'un répertoire des données de référence à accepter, nécessité d'applications spécifiques pour l'exploitation de celui-ci...

5. Analyse des systèmes existants

Les données peuvent être divisées comme suit :

- **les données jetables** – il faut les jeter

- **les données à conserver existant dans un format ouvert** tel que PDF ou PostScript ou pouvant aisément être transcrites dans l'un de ces formats.

Le coût de transcription doit être évalué.

5. Analyse des systèmes existants

- **les données à conserver existant dans un format propriétaire difficile à transcrire en format ouvert**.

Ces données peuvent nécessiter le maintien d'applications propriétaires spécifiques ; le coût de celles-ci doit être évalué, ainsi que le nombre de copies nécessaires sur la base de la fréquence d'utilisation (par exemple, si l'information est rarement utilisée, une seule copie sur une machine centralisée peut suffire).

Il peut également être nécessaire de maintenir un matériel spécifique pour utiliser ces applications.

5. Analyse des systèmes existants

3- Contraintes de sécurité

- Quel est le système actuel d'assignation des noms d'utilisateurs et mots de passe ? Existe-t-il une structuration des noms d'utilisateurs ? Laquelle ? Quelle est la politique d'expiration des mots de passe ?

- Certains systèmes nécessitent-ils une authentification supérieure à un couple utilisateur/mot de passe ?

5. Analyse des systèmes existants

3- Contraintes de sécurité

- Quelles sont les règles administratives et gouvernementales d'utilisation des ordinateurs (par exemple, existe-t-il des restrictions quant à l'utilisation d'Internet et du courriel) ?

- Y a-t-il des dispositions sécuritaires qui nécessitent l'utilisation de matériel ou de logiciel spécifique ?

6. Évaluer un cas concret et détaillé de migration

Ce cas de migration détaillée sera fondé sur les données rassemblées dans les étapes précédentes et comportera des points financiers importants.

6. Évaluer un cas concret et détaillé de migration

- Le TCO de l'environnement existant pour une période de temps raisonnable (3 à 5 ans par exemple).
- Le TCO d'environnements différents et le coût de migration vers chacun pour la même période.
- Les forces et faiblesses de l'environnement existant et de chaque alternative.

7. Consulter les utilisateurs

Assurer la communication, la sensibilisation et la documentation

7. Consulter les utilisateurs

- Leur **expliquer le raisonnement** sous-jacent à la migration et les effets qui leur seront perceptibles. Prendre sérieusement en compte leurs besoins et leur permettre de « jouer » avec les nouveautés aussi tôt que possible.

- Plus tôt les utilisateurs sont impliqués, mieux la migration est acceptée. Il peut y avoir des nécessités légales dans certains pays mais cela doit être fait dans tous les cas pour faciliter l'introduction de ce qui peut constituer une modification significative dans le travail quotidien.

7. Consulter les utilisateurs

- Créer **un pôle d'assistance** qui puisse répondre aux besoins des utilisateurs. Plus tard, lorsque la migration est en cours, celui-ci pourra résoudre les problèmes et devenir un centre d'excellence et de bonnes pratiques.

- Créer **un site intranet** avec une section « astuces et guides » qui puisse être mise à jour par les utilisateurs eux-mêmes. Il est important que les utilisateurs se sentent impliqués et le site donnera aussi au pôle d'assistance une idée du genre de problèmes auxquels sont confrontés les utilisateurs.

8. Démarrer un projet pilote

8. Démarrer un projet pilote

En supposant réalisé le cas concret, commencer avec des projets pilotes à petite échelle, de préférence dans un environnement isolé avec un petit nombre d'utilisateurs. Ceux-ci fourniront notamment :

- Des données pour affiner les modèles de TCO
- Des réactions d'utilisateurs qui permettront de faciliter l'introduction d'autres systèmes
- La validation ou la modification de l'architecture cible et du cas concret précédemment réalisé

9. Définir la vitesse de migration

Il est important d'avoir une stratégie de transition permettant aux systèmes qui doivent cohabiter de fonctionner simultanément afin que les activités de production puissent continuer correctement durant la période de migration.

9. Définir la vitesse de migration

Le passage en force : Pour les petites organisations

Tous les utilisateurs passent au nouveau système le même jour. Dans la pratique, il est vraisemblable que la migration soit planifiée sur une fin de semaine ou durant les vacances.

L'avantage réside dans l'absence de structures de double accès et de double maintenance par les équipes techniques.

Les inconvénients résident dans un niveau de risque très élevé et dans la très forte mobilisation de ressources durant la migration. Ce schéma semble ne pouvoir être appliqué que

9. Définir la vitesse de migration

Le passage en force : Pour les petites organisations

Dans la mesure du possible, ÉVITEZ LA MIGRATION EN FORCE.

Ce type de migrations nécessite le contrôle de tant de variables qu'elles échouent en général. Si c'est le cas, ce n'est pas, en général, du fait d'une défaillance du logiciel libre mais de celle du management.

9. Définir la vitesse de migration

La transition progressive par groupes

Les utilisateurs migrent par groupes. On fait migrer en général des groupes fonctionnels complets afin de minimiser les partages de données et les difficultés de travail de groupe.

Les risques peuvent être limités et les ressources gérées par l'adéquation de la taille des groupes.

9. Définir la vitesse de migration

La Transition individuelle : Projets pilote

Similaire à la transition progressive par groupes mais avec une taille de groupe d'une personne.

Cette méthode au goutte-à-goutte mobilise peu de ressources mais elle est peu efficace et sans doute peu adaptée aux grandes organisations. Elle peut cependant être appropriée aux projet pilotes.

10. Réaliser la migration de l'ensemble de l'organisation

10. Réaliser la migration

Déploiement total du projet conformément au choix retenus.

Cette phase implique une formation ultérieure de tous les utilisateurs et de l'équipe technique.

Processus de Migration

11. Faire un suivi des retours utilisateurs et résoudre les problèmes

11. Faire un suivi des retours utilisateurs et résoudre les problèmes

Certains besoins utilisateurs peuvent être suffisamment obscurs pour n'avoir pas été prévus à l'avance ni découvert durant les phases pilotes.

S'assurer que des ressources suffisantes restent disponibles pour répondre à ce type de besoins après la transition (support, hotline).

Gestion de la migration

ubuntu

Gestion de la migration

Les critères humains

Réussir l'intégration

Les Critères Humains

1. La peur de l'inconnu
2. L'effet de dilution du CV
3. La perte du pouvoir

1. La peur de l'inconnu

1. La peur de l'inconnu

Le comportement par défaut

L'utilisation du logiciel libre sera une nouveauté absolue pour de nombreux utilisateurs et administrateurs système. La peur de l'inconnu, naturelle, les fera résister à l'introduction de du logiciel libre en raison de sa nouveauté.

Les utilisateurs curieux

Certains utilisateurs, naturellement plus curieux, pourront être très volontaires pour tester les nouveautés et ce sont ceux à qui présenter le logiciel libre en premier.

1. La peur de l'inconnu

Les utilisateurs sortis de la réserve

L'expérience montre qu'une fois sortis de leur réserve, les gens trouvent qu'il n'est pas significativement différent d'utiliser le logiciel libre que le logiciel propriétaire.

Il est ainsi vraisemblable que le groupe initial d'utilisateurs passera au logiciel libre avec enthousiasme. Ce sont eux aussi qui fourniront sans doute les retours les plus utiles.

1. La peur de l'inconnu

Les utilisateurs sortis de la réserve

Ce premier groupe peut être utilisé dans les essais pilotes et, une fois expérimenté, peut contribuer à accompagner et éduquer les autres utilisateurs.

Les utilisateurs réservés

Dans la seconde phase, les utilisateurs qui resteront plus réservés nécessiteront un support plus avancé sous la forme de pôles d'assistance, intranets et utilisateurs locaux expérimentés.

1. La peur de l'inconnu

Les administrateurs système

Le même processus peut être utilisé pour les administrateurs système mais le niveau de formation sera significatif si l'environnement propriétaire existant n'est pas de type UNIX. L'équipe système en particulier doit voir ses peurs évacuées à une étape précoce.

Ils deviendront un point focal pour tous les problèmes qui surviendront et s'ils ne croient pas en le projet, ne seront pas en mesure d'encourager les utilisateurs de manière positive.

2. L'effet de dilution du CV

2. L'effet de dilution du CV

Utilisateurs ordinaires et administrateurs

Aussi bien l'équipe système que les utilisateurs peuvent penser que la non-utilisation de logiciels « standard » risque d'altérer leur capacité de développement de carrière.

Ce problème complexe nécessite une gestion prudente. Sans se sur-outiller dans cette approche, ce problème peut survenir assez souvent jusqu'à une adoption massive du logiciel libre.

3. La perte du pouvoir

Connaissance = Pouvoir

La perte du pouvoir

Les personnes qui connaissent les systèmes existants et leur paramétrage disposent d'un certain pouvoir et peuvent être très réfractaires à leur abandon si l'environnement logiciel libre est très différent de l'existant.

Cela aussi nécessite une gestion prudente car ceux-ci jouent un rôle critique dans l'exploitation des systèmes existants. Il peut être nécessaire de les intégrer aux premières formations afin de leur permettre de maintenir leur position dans l'organisation.

Réussir l'intégration

1. Introduire les nouvelles applications dans un environnement familier
2. Commencer par le plus simple
3. Anticiper les complexités futures

ubuntu

1. Introduire les nouvelles applications dans un environnement familier

De nombreuses applications logiciel libre fonctionnent sur des systèmes d'exploitation propriétaires et cela offre une opportunité d'introduction de celles-ci sans modification intégrale de l'environnement (OpenOffice.org/Ms-Office, Mozilla/MS-Internet Explorer et Apache/MS-IIS en environnement MS Windows).

1. Introduire les nouvelles applications dans un environnement familier

En plus d'être moins intrusive, cette approche permet de jauger les réactions des utilisateurs à une échelle réduite et la planification de la formation peut être fondée sur l'expérience réelle.

De plus, des problèmes tels que la conversion de formats de fichiers, macros et modèles peuvent être simplifiés si l'ancienne application reste provisoirement disponible.

2. Commencer par le plus simple

Les serveurs

Les premières modifications à effectuer sont celles qui ne perturbent pas la population utilisatrice, c'est-à-dire d'abord sur les serveurs.

Ceux-ci fourniront ainsi une plate-forme pour la modification ultérieure des clients. De nombreuses modifications serveurs seront compatibles avec l'environnement existant, ainsi l'effet perturbant sera minimisé.

2. Commencer par le plus simple

Les serveurs

Par exemple, les serveurs de noms DNS, les serveurs DHCP et les serveurs de bases de données à moteurs propriétaires tels qu'Oracle sont tous candidats à leur remplacement par leur équivalent logiciel libre et interopèrent avec le reste du système existant comme auparavant.

Certaines applications comme Samba ne seront pas utilisées dans un environnement purement logiciel libre mais permettent la coexistence de l'ancien environnement propriétaire et du logiciel libre. L'utilisation précoce de ces

3. Anticiper les complexités futures

Navigateurs & sites web

Imposer que le contenu produit par développement web maison ou par des prestataires soit visualisable sur tous les navigateurs actuels, en particulier les navigateurs libres.

Il s'agit dans tous les cas d'une bonne pratique aucun logiciel spécifique ne doit être nécessaire pour accéder au contenu en ligne d'une organisation. Des outils tels que **weblint** facilitent le contrôle de compatibilité des pages web.

3. Anticiper les complexités futures

Macros et scripts divers

Décourager la prolifération incontrôlée des macros et scripts dans les documents et feuilles de calculs ; chercher d'autres moyens de réaliser la fonctionnalité.

Cela est aussi une bonne pratique car l'utilisation de ceux-ci est un biais classique d'infection par les virus. De même, les macros peuvent facilement être utilisées pour dérober des informations et détourner des documents *(par exemple, elles permettent que le document affiche des éléments différents à l'écran et à l'impression)*.

3. Anticiper les complexités futures

Formats de fichiers ouverts standard

Imposer l'utilisation des formats de fichiers ouverts standard (ex. PostScript et PDF).

Il faut éviter d'utiliser des formats propriétaires pour les fichiers prévus pour la lecture et non la modification par le destinataire.

3. Anticiper les complexités futures

Formats de fichiers propriétaires

Lors de la rédaction de documents à plusieurs, utiliser un format à la plus petite version.

Par exemple, préférer le format MS-Word 2000 au format MS-Word 2007.
Cela augmente la probabilité de participation avec des applications libres.

3. Anticiper les complexités futures

Modèle de développement 3-tiers

Développer des systèmes fondés au minimum sur un modèle 3-tiers dans lequel le code applicatif est indépendant de l'interface homme-machine et des méthodes d'accès aux données.
Imposer par exemple, si possible, que l'interface fonctionne avec un navigateur libre.

Le développement d'applications selon ce principe modulaire en facilite la migration élément par élément.

3. Anticiper les complexités futures

Portabilité des programmes

Imposer que tout nouveau développement soit portable. Cela implique l'utilisation de langages portables standardisés tels que le C ANSI, Java, Python, Perl...

Éviter les langages et API spécifiques. Éviter de développer des applications nécessitant la présence d'autres applications propriétaires.

3. Anticiper les complexités futures

Portabilité des programmes

Eloigner les utilisateurs des outils de courriel qui utilisent des formats de stockage et communiquent avec les serveurs à l'aide de protocoles propriétaires.

De nombreuses applications stockent le courriel à l'aide d'IMAP. Si possible, trouver un moyen de stocker les carnets d'adresses et les informations de calendrier dans un format ouvert.

Quelques éléments techniques pour la migration

ubuntu

Elements techniques

1. Le poste de travail

2. Les serveurs

1. Le poste de travail

Le système de base

- Choix d'une distribution GNU/Linux (ex. Ubuntu)
- Environnement de bureau Gnome ou KDE
- Utilisateurs sans pouvoir
- Adressage réseau dynamique ou statique
- Possibilité d'authentification dans une base LDAP centrale

1. Le poste de travail

Bureautique

- OpenOffice.org est privilégié
- Existe pour des OS propriétaires et permet un usage préalable avant la découverte de Linux
- Une des meilleures interprétation des formats MS-Office

1. Le poste de travail

Le couriel

- Agents utilisateur (MUA) possèdent une interface proche de MS-Outlook et sont faciles d'apprentissage (ex. Evolution, Thunderbird)
- Autres logiciels (Kmail,...)
- Possibilité d'utiliser une solution web de travail collaboratif pour un moment.

1. Le poste de travail

Accès web

- Navigateur Mozilla Firefox
- Autres navigateurs selon les fonctionnalités voulues

1. Le poste de travail

Autres applications

- Agenda personnel et gestion de contacts (Evolution, Kmail)
- Groupwares, travail collaboratif (phpGroupWare, Open-Xchange, Zimbra)
- Sécurité : clients VPN, antivirus, pare-feux,

2. Les serveurs

Le système de base

- Choix d'une distribution GNU/Linux (ex. Debian)
- Choix de version stable
- Possibilité d'insérer d'autres seveurs (ex. OpenBSD comme pare-feux)

2. Les serveurs

Courriel

- Agents de transport de courrier MTA (Exim, Postfix...)
- Agents de stockage (Courier-IMAP, Cyrus...)
- Intégration Antispams (Spamassassin)
- Intégration Antivirus (ClamAV)

2. Les serveurs

Services web

- Choix de Apache qui est le plus performant des serveurs web actuels
- Possibilité d'association avec d'autres services pour des tâches spécifiques
- Jakarta est un sous projet utilisation java côté serveur
- Zope conçu pour fournir un support de contenu web dynamique, est fondé sur un modèle orienté objet

2. Les serveurs

Bases de données et annuaire

- Pour les bases de données à accès principal en lecture seule, MySQL
- Pour des grandes de bases de données, PostgreSQL
- Annuaire LDAP (OpenLDAP)

2. Les serveurs

Autres applications

- Serveurs d'infrastrucrures réseau : Routage, DNS, DHCP
- Serveurs de fichiers : NFS, Samba
- Sécurité : Firewall, Proxy, Serveur VPN
- Surveillance réseau : SNMP, (Nagios, MRTG)
- Impression : CUPS, LPRng
- Sauvegarde et restauration : Dump, Restore, (Amanda, BackupPC)