1 Remerciements

À Unipress et ses employés pour leur sympathie et pour m'avoir accepté et intégré dans leur environnement de travail.

À Stéphan Pozzi pour sa relecture du présent document et ses conseils.

À Matthieu Vanderdonck, gradué de l'IPL en 2005 pour sa relecture du présent document et ses conseils.

À Philippe Léonard pour sa relecture du présent document.

À Michèle Denis pour sa relecture du présent document.

À la communauté du logiciel libre pour m'avoir rendu ces technologies accessibles et un remerciement tout particulier aux personnes actives du forum qtfr.org pour avoir tenté d'apporter des réponses à mes questions.

Table des matières

2 Introduction

Dans le cadre de mon stage de fin d'étude, et en réponse aux besoins de la société Unipress en matière de surveillance à distance de serveurs de virtualisation Xen, il m'a été demandé de prendre en charge toutes les étapes de production d'un logiciel pouvant répondre à ces besoins.

2.1 Utilité de la virtualisation

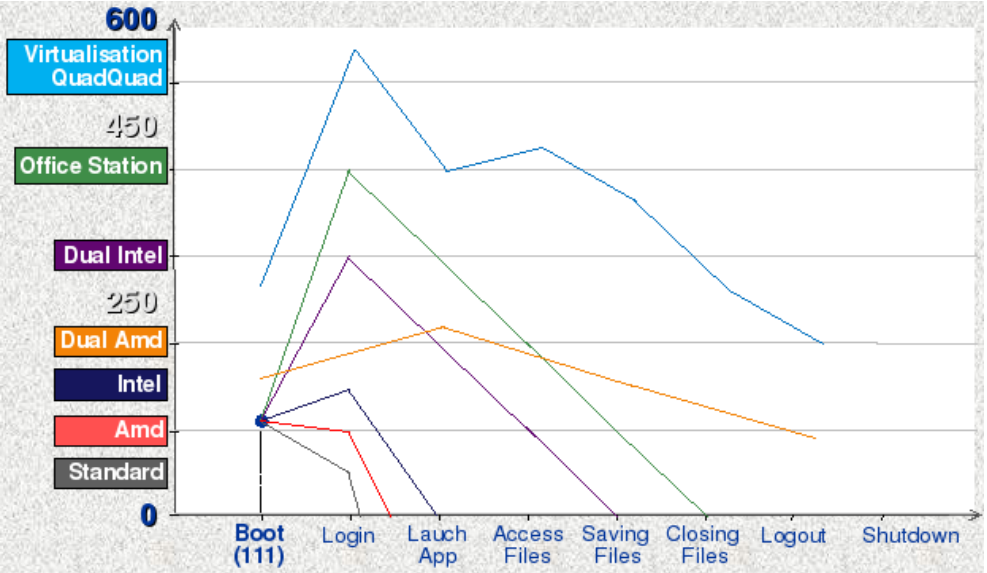
Que ce soit sur un serveur ou sur une machine de travail, il arrive souvent que les ressources de la machine ne soient pas tout le temps exploitées comme elles le devraient. Sur une journée d'utilisation d'une machine de travail, il n'est pas rare de constater que la machine en question soit au repos durant la majorité du temps, et que son utilisation ne provoque que des pics succincts d'utilisation des ressources. Ces pics représentent par exemple le démarrage de la machine ou l'ouverture d'une grosse feuille de calcul, ils se traduisent au niveau de l'utilisateur par une diminution de la réactivité de la machine mais ne sont pas fréquents dans une journée de travail. Une fois la feuille de calcul chargée, l'utilisateur passera probablement plusieurs dizaines de minutes à effectuer des opérations sur ce document, exploitant donc peu les ressources de la machine. Si une machine plus puissante était utilisée, ces pics d'utilisation pourraient être réduits, mais les coûts d'achats de machines plus puissantes seraient également plus élevés. La virtualisation permet de faire cohabiter plusieurs machines virtuelles pouvant tourner sous des systèmes d'exploitation différents sur une même machine physique et d'ainsi répartir ces pics d'utilisation sur une machine plus puissante.
Ainsi, nous pouvons par exemple imaginer de faire tourner simultanément sur le même serveur physique, un serveur de messagerie sous GNU/Linux, et des postes utilisateurs sous Windows XP accessibles au moyen de terminaux au travers du réseau. Si cela présente des avantages évidents à court terme, cela en présente également à long terme :

Réduction des coûts d'achats des machines Acheter un serveur accompagné de terminaux légers revient moins cher que d'acheter des stations de travail.

Réduction de la consommation électrique L'énergie est une denrée qui devient de plus en plus rare et chère, en plus des économies évidentes sur la facture d'électricité, l'installation est également plus écologique.

Impact managérial L'utilisation de la virtualisation occasionnant une diminution du nombre de machines à maintenir, leur coût de maintien total se verra diminué.

FIG. 1 – Utilisation des ressources d'une machine



Utilisation optimale des ressources d'un parc de machines La virtualisation permet de répartir les machines virtuelles sur les machines physiques en fonction de leurs charges respectives.

Réduction de la place utilisée par les machines N'utiliser plus qu'un seul serveur physique avec un système de virtualisation permet d'économiser de la place par rapport à un système constitué de plusieurs serveurs physiques. De même, dans le cas d'un système constitué de terminaux légers connectés au réseau, la place prise par un de ces terminaux est ridicule comparée à une station de travail.

Réduction de la dissipation thermique Les opérations de calcul étant effectuées sur le serveur et non plus sur des stations de travail, la dissipation thermique dans l'environnement de travail des utilisateurs finaux est limitée à celle des terminaux légers et des périphériques et se voit par conséquent fortement réduite. Les serveurs se situant en général dans une pièce leur étant dédiée, ceux-ci peuvent bénéficier d'un système de refroidissement plus efficace et plus bruyant sans répercussion sur l'environnement de travail des utilisateurs finaux.

Augmentation des performances réseau Les performances des communications réseau entre le serveur de messagerie et les postes clients au sein d'une machine virtualisante ne sont plus limitées par les performances d'un réseau mais par le bus interne de la machine. Afin d'obtenir des conditions de travail acceptables dans le cas d'une utilisation avec des terminaux légers, il est cependant nécessaire d'utiliser un réseau à

faible latence, voire même à haut débit (1 Gbps) à partir d'un certain nombre de terminaux.

Diminution des coûts des licences logicielles Dans le cas d'une utilisation de logiciels nécessitant l'obtention d'une licence d'utilisation, plusieurs cas concrets ont prouvé que l'utilisation d'un logiciel de manière virtualisée sur une même machine physique ne nécessitait l'obtention que d'une seule licence d'utilisation.

Dimensionnement dynamique de l'environnement d'une application Les machines virtuelles pouvant être modifiées à souhait, l'architecture informatique globale devient dynamique et facilement adaptable aux évolutions des besoins de certaines applications.

2.2 Xen et ses outils

La virtualisation est la tendance actuelle dans le monde de l'informatique. Ses avantages sont nombreux : économie d'énergie, économie de matériel, facilité d'administration, sécurité, ... Parmi les solutions du marché se trouve Xen, une solution libre de virtualisation à base d'hyperviseur, supportant également depuis la version 3 la virtualisation complète. Xen est fourni par défaut avec un utilitaire en ligne de commande (xm) qui permet d'effectuer toutes les tâches que Xen peut mener. Cependant, ces commandes peuvent vite paraître fastidieuses à utiliser pour un utilisateur non initié. De plus, ces commandes doivent être effectuées directement sur la machine hôte et ne permettent donc pas une vue globale de plusieurs machines virtualisantes. Il existe de nombreux projets destinés à rendre la gestion de Xen plus pratique via de multiples interfaces graphiques. Cependant, beaucoup d'entre-elles sont propriétaires et payantes (telle que Zenworks), et bien qu'il existe des solutions libres (telles que Enomalism, Virt-manager et Xenman), elles sont limitées à un environnement, trop compliquées à utiliser pour un utilisateur final, nécessitent l'utilisation d'un navigateur Internet, ne permettent pas une gestion native au travers du réseau ou ne présentent tout simplement pas toutes les fonctionnalités attendues.

2.3 Le projet

En réponse à ce besoin, la société Unipress m'a donc demandé d'élaborer une interface de gestion pour Xen correspondant à ces critères :

- Portabilité de l'application (GNU/Linux, Windows et éventuellement MacOS),

- Application indépendante, ne nécessitant pas un serveur pour la faire fonctionner,
- Prise en main simple pour un utilisateur non initié et interface intuitive,
- Possibilité de gestion simultanée de plusieurs hôtes,
- Gestion des actions de base pour les machines virtuelles (démarrage, arrêt, redémarrage, arrêt brutal),
- Surveillance graphique des différents domaines,
- Utilisation de la nouvelle API Xen XML-RPC,
- Fonctionnalité de « migration en direct ».

Après analyse des besoins, 2 valeurs sûres dans le monde du développement du logiciel libre ont été désignées afin de mener à bien ce projet : Python 2.5 et Qt 4. XenMonitor ont donc été developpés grâce à ces 2 technologies.

N'ayant jamais approché le monde de la virtualisation avec Xen, le langage Python et le framework Qt avant mon stage, celui-ci a davantage été pour moi une expérience d'apprentissage qu'une expérience d'analyse et de production d'un logiciel. Bien que ce deuxième aspect ait été présent dans mon stage, le nombre de nouvelles connaissances à assimiler a laissé une place plus petite à la réalisation d'un réel « produit fini », à condition qu'on puisse qualifier un logiciel ainsi. Mais après tout, la seule rémunération que l'on perçoit dans un stage est ce que l'on y apprend. Les technologies que j'ai pu approcher durant mon stage m'ont passionné et je dois bien avouer que même si j'ai passé une partie de mon temps libre à me documenter à propos de celles-ci, je suis loin d'en avoir fait le tour. Ce mémoire est donc à l'image de mon stage et concerne davantage les orientations technologiques de Xen et les avantages de l'utisation de Python et de Qt que la réalisation d'un logiciel.

3 Présentation du contexte

3.1 Société

Spécialisée dans l'assemblage de serveurs informatiques sur mesure depuis 1992, la société Unipress emploie une dizaine de personnes. Sa petite taille et son dynamisme lui permettent de tenir tête à ses concurrents, des géants de l'informatique tels que Dell ou HP en offrant la possibilité de pouvoir fournir à ses clients un service personnalisé. Une machine qui correspond exactement à leurs besoins et l'utilisation des dernières technologies du moment est une réelle force qui est appréciée par ceux-ci. Unipress possède en effet de nombreux clients de taille respectable comme des universités, des villes, des infrastructures gouvernementales, des datacenters ou encore des fermes de rendu graphique partout en Europe.

Parallèlement à la vente de serveurs, Unipress et ses collaborateurs fournissent également du service aux clients qui le désirent afin par exemple d'assurer l'intégration d'une nouvelle machine au sein d'un système existant, d'y déployer des solutions logicielles ou encore d'effectuer une migration de l'ancien parc informatique vers un nouveau. Parmi les services proposés par Unipress, on retrouve des offres basées sur la virtualisation, avec ou sans redondance, ou éventuellement avec un stockage externe des images disques des machines virtuelles.

La solution de virtualisation qu'Unipress met en place chez ses clients est basée sur GNU/Linux et Xen. L'intégration des nouvelles versions de Xen dans les distributions GNU/Linux étant en constante évolution, le responsable technique d'Unipress préfère ne pas se lier avec une distribution spécifique. Une grande partie des solutions logicielles déployées par Unipress étant du logiciel libre, l'essentiel du travail consiste donc à se tenir régulièrement informé des évolutions des diverses solutions. Cette facette du travail pourrait être qualifiée de « veille technologique ».

3.2 Environnement informatique

Matériel Un serveur a été mis à ma disposition afin d'y reproduire un environnement de test avec OpenSUSE 10.3 et Xen 3.1. En ce qui concerne le développement, il a été effectué sur mon ordinateur portable sous Debian, qui me servait également de temps en temps de serveur de test d'appoint avec Xen 3.2. Par la suite, un deuxième serveur de test installé de la même manière que le premier a également été mis à ma disposition.

Langage Python 2.5 accompagné du framework Qt 4.3 et de son interface pour Python. La librairie de dessin de graphiques Qwt 5 ainsi que son interface pour Python ont eux aussi été utilisés.
Qt 4.4 étant sorti vers la fin de mon stage, je l'ai également utilisé afin de tester le bon fonctionnement de l'application avec cette nouvelle version du framework. Je n'ai cependant pas exploité ses nouvelles fonctionnalités lors du développement.

Outils

Epydoc : Aide à la génération de documentation.
Eric 4 : Environnement de développement intégré spécialisé entre autre pour Python et Qt, écrit en Python et Qt.
PyChecker : Détection de bugs courants.
PyLint : Amélioration de la qualité du code Python : suppression des imports et des variables non utilisés, vérification de la syntaxe, des conventions de nommage, ...
Python 2.5 : Interpréteur Python.
PyUic 4 : Compilateur d'interface Qt 4 pour Python.
Qt Assistant 4 : Documentation Qt.
Qt Designer 4 : Aide à la conception d'interfaces graphiques.
Qt Linguist 4 : Aide à la traduction de l'application.
Umbrello : Outil de modélisation UML.

Latex et Kile, une interface utilisant ce dernier ont également été utilisés pour la rédaction du présent document ainsi que du manuel. Tous ces logiciels sont des logiciels libres.

4 Calendrier

4.1 Semaine 1 : du 18 au 22 février 2008

- Découverte de l'entreprise, de son personnel, de ses activités ainsi que de son mode de fonctionnement.
- Installation de l'environnement de développement et du premier serveur de test.
- Découverte du système de clients légers nComputing.
- Découverte de Xen et de son API.
- Choix des technologies utilisées pour le développement.
- Découverte des principaux outils de développement.
- Choix du cycle de développement.
- Première ébauche de l'interface principale.

4.2 Semaine 2 : du 25 au 29 février 2008

- Définition des classes et de la structure du logiciel.
- Tests de récupération de données avec l'API Xen.
- Amélioration de l'interface principale.
- Recherches autour du NAS Longshine LCS-8311.
- Déploiement de Xen comme deuxième serveur de test d'appoint sur l'environnement de développement.
- Tests en vue de l'implémentation de la gestion de multiples serveurs Xen.

4.3 Semaine 3 : du 3 au 7 mars 2008

- Implémentation de la gestion des déconnexions et reconnexions à un serveur.
- Amélioration de l'interface.
- Implémentation de la gestion de multiples serveurs.
- Mise en place de la gestion de l'internationalisation avec Qt.
- Premier retour sur le travail effectué.
- Suite de l'analyse (persistance des données ? centralisation de la surveillance ? graphiques ?).
- Gestion de l'affichage des VM.
- Fermeture des sessions à la fermeture de l'application.
- Implémentation de la fonctionnalité de démarrage/arrêt d'une VM.

4.4 Semaine 4 : du 10 au 14 mars 2008

- Tests de communication avec le serveur VNC intégré de Xen.
- Implémentation de la gestion des informations relatives aux interfaces réseau.
- Lecture des recommandations officielles Python (conventions de nommage, ...) et adaptation du code.
- Refactoring général et réorganisation des packages.

4.5 Semaine 5 : du 17 au 21 mars 2008

- Ajout d'une information détaillée à l'utilisateur lors d'une impossibilité de connexion à un serveur.
- Recherche d'une librairie de dessin de graphes avec Qt et choix de Qwt.
- Ajout de la fonctionnalité de redémarrage d'une VM.
- Implémentation de l'affichage des graphes d'utilisation des VCPUs sous forme de QDockWidgets.
- Utilisation de QDockWidgets pour les autres éléments de la fenêtre principale.

4.6 Semaine 6 : du 25 au 28 mars 2008

- Installation d'un serveur destiné à la production et résolution de problèmes.
- Amélioration du rendu visuel des fenêtres de graphes d'utilisation des VCPUs.
- Ajout du support du graphe d'utilisation des VCPUs pour les hôtes.

4.7 Semaine 7 : du 31 mars au 4 avril 2008

- Ajout du support multi-VCPUs pour les graphes d'utilisation des VCPUs.
- Recherches en vue de la correction d'un bug rendant impossible l'utilisation de Virt-manager suite à une mise-à-jour d'OpenSUSE[1].
- Mise-en-forme des boutons dans une barre d'outils.
- Ajout de la fonctionnalité permettant de tuer une machine virtuelle.
- Ajout d'une boîte de dialogue de gestion des préférences.

[1] Bug résolu par l'utilisation de la version précédente de plusieurs paquets relatifs à dbus. Le bug a été signalé sur le Bugzilla de Novell par la réouverture d'un ancien rapport de bug similaire (https ://bugzilla.novell.com/show_bug.cgi ?id=349892) mais persiste encore à ce jour dans OpenSUSE 10.3.

- Gestion manuelle des connexions/déconnexions des hôtes.
- Analyse de la problématique de l'extinction d'une machine virtuelle complètement virtualisée et test de WindowsXenPV 0.8.8.
- Ajout de la notion de « connexion en cours »au sein de la gestion des hôtes.

4.8 Semaine 8 : du 7 au 11 avril 2008

- Implémentation d'un système de persistance des préférences basé sur XML.
- Implémentation de la gestion des fichiers de traduction basé sur XML.
- Participation à la conférence organisée dans les locaux de la société par Hans de Lange portant sur les systèmes multi-utilisateurs, la virtualisation et le stockage de masse.
- Implémentation de la gestion des configurations d'hôte.

4.9 Semaine 9 : du 14 au 18 avril 2008

- Participation à une réunion sur l'organisation de la société.
- Fin de l'implémentation de la gestion des configurations d'hôte.
- Implémentation d'options de connexion/reconnexion automatique à un hôte.
- Amélioration du système de préférences.
- Test du logiciel sous Windows XP.

4.10 Semaine 10 : du 21 au 25 avril 2008

- Analyse de l'utilité d'un modèle de données centralisé pour la gestion des domaines.
- Implémentation d'un modèle de données pour le panneau des propriétés du domaine.
- Tests d'utilisation d'une image disque VMWare sur Xen.
- Installation d'une machine virtuelle Windows XP + système nComputing sur un serveur.
- Découverte de PyLint afin de « nettoyer » le code et d'améliorer sa qualité (suppression des imports et variables non-utilisés, vérification de la syntaxe, des conventions de nommage, ...).
- Amélioration de la gestion des erreurs.

4.11 Semaine 11 : du 28 au 30 avril 2008

- Découverte de PyChecker afin de détecter des bugs courants.
- Recherches à propos de l'utilisation d'un modèle de données centralisé pour la gestion des domaines.
- Début d'implémentation d'un deuxième modèle de données pour la vue principale.

4.12 Semaine 12 : du 5 au 9 mai 2008

- Utilisation d'une QResource afin de stocker les ressources utilisées par le logiciel (icônes, ...).
- Ajout d'une QDockWidget afin de gérer l'affichage du journal sur la fenêtre principale.
- Amélioration de la gestion de l'internationalisation afin de la rendre plus dynamique et suppression du fichier XML.
- Utilisation des QSettings pour la gestion des préférences à la place du système à base de fichier XML.
- Écriture de jeux de tests pour la gestion des préférences et de l'internationalisation.
- Implémentation de la persistance de l'état des QDockWidgets des propriétés du domaine et du journal.

4.13 Semaine 13 : du 13 au 16 mai 2008

- Découverte d'Epydoc et génération de documentation.
- Mise-en-forme des docstrings afin de satisfaire aux recommandations Python et à Epytext, le langage de balisage utilisé par Epydoc.
- Mise en place de WebSVN, publication des sources et lien sur Sourceforge.
- Utilisation de vues spécialisées pour les propriétés du domaine et le journal.
- Implémentation de la fonction de migration en direct d'une machine virtuelle.
- Gestion des échecs d'authentification de session et amélioration de la gestion des erreurs.
- Installation d'un deuxième serveur de test.
- Utilisation d'un modèle de données et d'un mapper pour la gestion des configurations d'hôte dans la boîte de dialogue des préférences.

4.14 Semaine 14 : du 19 au 23 mai 2008

- Implémentation d'un modèle centralisé pour la gestion des domaines.
- Correction de bugs.

Il va de soi qu'à cette liste sommaire des étapes les plus importantes du stage s'ajoutent de nombreuses heures de recherches et de lecture de documentations et de cours dont une liste non-exhaustive est disponible dans la bibliographie.

5 Intégration

5.1 L'empaquetage

L'empaquetage est une étape importante permettant de faciliter le déploiement d'une solution logicielle. Il est cependant important de préserver une certaine modularité dans l'empaquetage d'une application, notamment au niveau des fichiers de traduction (voir paragraphe sur l'internationalisation en 7.6). Sous GNU/Linux, la majorité des distributions incluent un système de déploiement de logiciels et de gestion de mises-à-jour ainsi que des outils permettant l'empaquetage de logiciels. Sous Windows, il existe des solutions indépendantes destinées à faciliter le déploiement de logiciels écrits en Python. Une de ces solutions se nomme Py2exe et permet de transformer des scripts Python en un seul fichier exécutable ne nécessitant pas d'installation de Python sur le système. XenMonitor étant encore en phase de test, je n'ai pas encore étudié ces solutions, mais elles seront probablement utilisées dans le futur de XenMonitor.

5.2 Distribution du logiciel

XenMonitor sera distribué sous licence GPLv3, tout comme l'est le framework Qt. Les sources peuvent être visualisées à l'adresse http ://svn.openminds.org/listing.php ?repname=XenMonitor ou récupérées directement à partir du dépôt Subversion de développement de cette manière :

```
svn co svn://open-minds.org/xenmonitor/trunk
```

Un espace pour le projet a également été créé sur Sourceforge, accessible à l'adresse http ://sourceforge.net/projects/xenmonitor. Le nom de domaine www.xenmonitor.com a également été réservé dans le but d'y héberger une page de présentation, des copies d'écrans, de permettre la récupération des sources du logiciel ou encore d'obtenir du support autour de celui-ci.

5.3 L'importance du logiciel libre

Afin d'illustrer l'importance du logiciel libre, je me permettrai d'emprunter une analogie faite par Richard Stallman, initiateur du projet GNU, entre un logiciel et une recette de cuisine. « Il existe de nombreux points communs entre un logiciel et une recette de cuisine : une liste d'étapes à suivre, des règles qui déterminent à quel moment vous avez fini ou comment revenir en arrière. A la fin, on obtient un certain résultat. Si vous aimez cuisiner, vous échangez sans doute vos recettes avec vos amis, et vous êtes probablement

amené à les modifier. Si le résultat vous plaît, et que vos amis s'en régalent, il y a des chances pour que vous leur donniez la nouvelle version de cette recette. Et maintenant, imaginez un monde ou vous ne pourriez pas changer votre recette parce que quelqu'un aurait décrété qu'il est impossible de la modifier. Et imaginez que si vous partagiez quand même votre recette avec vos amis, il vous traiterait de pirate et ferait tout pour vous envoyer en prison pendant des années... »

La notion de logiciel propriétaire, pourtant aujourd'hui fort répandue, n'est pas née avec l'informatique. A la base, l'informatique était une science et les informaticiens, comme les chercheurs, partageaient leurs connaissances afin de faire évoluer le savoir de l'Humain. Imaginez un seul instant que dans le secteur de la recherche scientifique on ait déposé autant de brevets qu'on l'a fait ces dernières années dans celui de l'informatique, nous serions probablement sous-développés dans beaucoup de domaines. La majorité des technologies innovantes d'aujourd'hui n'auraient pas pu voir le jour sans le logiciel libre et sans les idées qu'il véhicule, Internet en est le meilleur exemple. La licence publique générale GNU (GPL) est une licence libre, elle préserve 4 libertés fondamentales pour l'utilisateur :

- la liberté d'exécuter le programme, pour tous les usages,
- la liberté d'étudier le fonctionnement du programme, et de l'adapter à ses besoins,
- la liberté de redistribuer des copies, donc d'aider son voisin,
- la liberté d'améliorer le programme et de publier ses améliorations, pour en faire profiter toute la communauté.

5.4 Le logiciel libre et l'économie

Bien sûr, pour un projet de fin d'études, il est tout-à-fait envisageable que les travaux effectués par un étudiant soient publiés sous une licence libre et ne rapportent pas d'argent. Après tout, le but premier d'un étudiant est très proche de celui du scientifique : apprendre. Et l'apprentissage ne peut exister s'il n'y a pas de partage du savoir.

Cela dit, un amalgame est souvent fait entre logiciel libre et gratuité du logiciel. Si la GPL permet à n'importe quel utilisateur de redistribuer librement un logiciel acheté, on se rend vite compte que la notion d'achat de droits d'utilisation du logiciel est une notion dépassée. Le développement de logiciels ne représente en fait qu'une faible partie du marché de l'informatique, une des plus grandes étant la personnalisation des logiciels pour les

entreprises ainsi que le déploiement et l'intégration de ces technologies. Forte de ce constat, l'économie du logiciel libre ne se base pas sur la vente d'un logiciel ou de droits d'utilisation, mais sur le service que l'on peut proposer autour de celui-ci.

Le concept de logiciel libre, bien qu'inspiré par les débuts de l'histoire de l'informatique, présentait et présente encore une réticence de la part de nombreuses personnes. On ne peut pourtant faire que le constat suivant : l'utilisation des logiciels libres est en constante augmentation. Que ce soit chez le particulier avec le logiciel de transfert FTP FileZilla, le navigateur Internet Firefox, la suite bureautique OpenOffice, le lecteur multimédia VLC ou encore le logiciel de retouche d'images The Gimp, ou en entreprise avec le serveur de fichiers Samba, le serveur web Apache, le serveur d'E-mails Postfix, le serveur VPN OpenVPN ou encore le système d'exploitation GNU/Linux, le logiciel libre est partout, et l'est de plus en plus. Le nombre d'entreprises dont l'activité principale tourne autour du déploiement de solutions libres est également croissant, créant ainsi un nouveau modèle économique très différent de celui utilisé par le logiciel propriétaire. Cette augmentation prouve avant tout une chose : ce nouveau modèle économique est viable, présente de nombreux avantages et pourrait bien, à terme, remplacer celui du logiciel propriétaire. Les freins les plus efficaces dans l'évolution de l'utilisation du logiciel libre sont encore les apprioris et les habitudes des utilisateurs. Cependant, les mentalités évoluent et les migrations de masse se multiplient dans tous les secteurs.

5.5 Le logiciel libre et ses avantages

L'utilisation de logiciels libres présente de nombreux avantages, que ce soit pour l'utilisateur d'un logiciel ou pour son concepteur.

- L'utilisateur du logiciel n'a pas de dépendance malsaine envers le concepteur du logiciel. Si le prestataire de services fait mal son travail, l'utilisateur peut décider d'en changer. De même, si le concepteur du logiciel fait faillite et ne peut donc plus assurer la maintenance du logiciel, l'utilisateur peut également faire appel à un autre prestataire de services tout en gardant son installation.
- Si le logiciel présente un certain succès auprès des utilisateurs, sa distribution sera plus large et la société du concepteur du logiciel sera également plus largement connue dans son domaine. Sa réputation sera fonction de la qualité de son produit et elle sera susceptible d'être contactée par de nouveaux clients.

- Si un ou plusieurs utilisateurs du logiciel désirent voir apparaître une nouvelle fonctionnalité dans celui-ci mais que le concepteur n'est pas en mesure de l'implémenter, ou qu'il juge tout simplement que le nombre d'utilisateurs désirant cette fonctionnalité ne permet pas de rentabiliser son développement, ceux-ci peuvent faire appel à un prestataire de services afin d'implémenter la fonctionnalité désirée.
- Le logiciel libre fait profiter l'économie locale d'un pays : au lieu d'acheter des droits d'utilisation dans un pays, l'utilisateur fera appel aux services d'un prestataire local.
- Partant du principe que des sociétés gagnent de l'argent grâce au déploiement et à la personnalisation d'un logiciel libre en entreprise, ceux-ci peuvent décider de faire évoluer ledit logiciel afin de présenter un meilleur service à leurs clients. Le concepteur original du logiciel pourra ainsi bénéficier de ces améliorations et vendre le service approprié à la nouvelle version.
- Si le logiciel présente un certain succès auprès des utilisateurs, il est possible qu'une communauté se forme autour de celui-ci, pouvant apporter des avis, des idées, des améliorations, des traductions, ou encore de la documentation, créant ainsi un groupe de travail collaboratif centré sur le logiciel.

5.6 Xen et sa position par rapport aux solutions du marché

On distingue pour le moment 3 techniques de virtualisation :

La virtualisation par isolation Il s'agit de la technique de virtualisation la plus simple, le système hôte gère des zones dans le système, cloisonnées tant au niveau de leurs processus que de leur arborescence. Seule la zone principale n'est pas limitée. Cette solution permet une certaine facilité dans le partage des ressources disques et réseau avec la zone principale mais ne permet pas l'utilisation d'un système d'exploitation différent du système hôte. En fonction de la solution utilisée afin de mettre en place la technique, l'isolation est plus ou moins bonne mais n'est pas parfaite. Cette technique est cependant très performante étant donné qu'elle n'utilise ni empilage, ni logiciel de virtualisation. Cette technique est utilisée par Sun pour ses conteneurs Solaris. Elle peut également être mise en place sur un système GNU/Linux grâce à OpenVZ, Linux-VServer ou encore un simple chroot. Les systèmes BSD permettent également la mise-en-place de ce type de solution grâce aux jails BSD.

La virtualisation complète ou partielle Ce type de solution permet de s'affranchir du matériel en émulant totalement ou partiellement celui-ci et en présentant au système invité un BIOS logiciel. L'avantage consiste en le fait que le système invité n'a pas conscience qu'il est virtualisé et ne doit donc pas être adapté s'il n'a pas été conçu pour être virtualisé. Cette technique nécessite donc un système hôte afin de faire fonctionner la solution, cependant cet empilage occasionne des pertes de performances pouvant être assez gênantes (au niveau des accès disques ou réseau par exemple), voire même handicapantes si le processeur est également émulé. Cette technique est utilisée par VirtualBox, VMWare, QEMU ou encore KVM.

La virtualisation par hyperviseur Solution nettement plus performante que la virtualisation complète, celle-ci nécessite que le système d'exploitation invité puisse dialoguer directement avec l'hyperviseur (une sorte de chef d'orchestre chargé de répondre aux différentes demandes de ressources effectuées par les machines virtuelles) et ait donc conscience qu'il est virtualisé, on parle ici de paravirtualisation. Cette technologie est utilisée par Xen et nécessite une adaptation du système invité afin qu'il puisse être exécuté avec un niveau de privilège inférieur à celui de l'hyperviseur, ce qui n'est pas toujours possible avec des systèmes comme Windows dont le code n'est pas modifiable. Il est à noter que Xen à partir de sa version 3 permet de gérer les systèmes d'exploitation invités qui ne supporteraient pas l'hyperviseur en les virtualisant complètement. Bien que VMWare ESX fasse de la virtualisation complète, il utilise un noyau léger nommé vmkernel (empilage léger) au lieu d'un système d'exploitation hôte comme le fait VMWare, l'architecture de Xen et de VMWare ESX sont donc très similaires.

Bien que la virtualisation complète soit la technique la moins performante, l'utilisation de technologies de support matériel de la virtualisation, telles que AMD-V (AMD Virtualization, anciennement Pacifica) et Intel VT (Virtualization Technology, anciennement Vanderpool), comble les lacunes du jeu d'instruction x86 en matière de virtualisation en permettant de virtualiser les accès mémoire et de faciliter le partage de temps de calcul entre les systèmes virtualisés au sein même du processeur. Ces technologies permettent une simplification de la complexité du logiciel de virtualisation et une réduction de la dégradation des performances. Le support de ces deux technologies est intégré dans Xen au sein d'une couche nommée HVM (Hardware Virtual Machine), Xen requiert d'ailleurs la présence de l'une d'elle sur le système afin d'effectuer de la virtualisation complète.

5.7 Quelques notions à propos de Xen

Afin de mieux comprendre le contenu de ce document, quelques notions à propos de Xen et de son fonctionnement sont nécessaires. Xen est donc un hyperviseur capable de virtualisation complète. Bien qu'il soit également opérationel avec NetBSD, FreeBSD, Plan9 et GNU/Hurd, on le retrouve le plus souvent au sein d'une distribution GNU/Linux. Le noyau de cette distribution est modifié afin d'intégrer l'hyperviseur. Le système une fois démarré est appelé Dom0. Xen jouera ensuite le chef d'orchestre afin de partager au mieux les ressources de la machine entre les demandes du Dom0 et celles des DomU, les machines virtuelles. Dans le cas de la paravirtualisation d'un système GNU/Linux, le noyau du système invité est également modifié afin qu'il puisse être conscient du fait qu'il est paravirtualisé. La modification apportée au noyau n'est cependant pas la même dans le cas d'un Dom0 ou d'un DomU. Bien que les modifications apportées au noyau afin d'utiliser le système en tant qu'invité fassent partie intégrante de la branche de développement principale du noyau Linux depuis la version 2.6.23, ce n'est pas encore le cas des modifications permettant l'intégration de l'hyperviseur. Pour ce qui est des systèmes qui ne peuvent pas être paravirtualisés, une virtualisation complète sera utilisée comme alternative à la paravirtualisation. En ce qui concerne les ressources, pour la mémoire, Xen permet l'allocation d'une certaine quantité de mémoire à un DomU, la mémoire restante étant allouée au Dom0. En ce qui concerne les processeurs, Xen permet d'allouer une certaine quantité de processeurs virtuels (VCPUs) à un domaine. Le nombre maximum de VCPUs attribués à un domaine ne peut pas dépasser le nombre de processeurs physiques que la machine contient, il est cependant possible d'avoir un nombre total de VCPUs alloués supérieur au nombre de processeurs physiques de la machine, laissant à l'hyperviseur le soin de gérer les différentes demandes de ressources des machines virtuelles et de les attribuer aux processeurs physiques disponibles.

5.8 Exemple d'utilisation de Xen en production

La société Lexitech est spécialisée dans la traduction. Son parc informatique était composé d'une vingtaine de machines sous Windows XP connectées en réseau ainsi que d'un serveur d'E-mails Exchange sous Windows 2003. Dans un premier temps, l'utilisation de la virtualisation n'était pas totalement pertinente, une machine sous Windows XP pouvant être utilisée par plusieurs utilisateurs simultanément au moyen d'une solution de terminaux légers tels que nComputing et le service d'E-mails et de stockage de données pouvant être assuré par une seconde machine sous GNU/Linux. Cependant,

la société Lexitech désire également utiliser project-open, une solution libre et orientée web de gestion de projets pour petites et moyennes sociétés. project-open peut être installé sur n'importe quelle machine munie d'un serveur web, de Tcl et des dépendances nécessaires, le déploiement pouvant être plus ou moins complexe selon l'environnement cible choisi. Cependant, il existe également sous forme de machine virtuelle VMWare dans laquelle est déployée une distribution GNU/Linux OpenSUSE ainsi qu'un serveur web, project-open et tout ce dont il a besoin pour fonctionner. L'utilisation de la virtualisation prend maintenant tout son sens avec :

- une machine virtuelle contenant un Windows XP utilisé par plusieurs utilisateurs au moyen de terminaux légers,
- une machine virtuelle contenant la solution project-open déployée sur une distribution GNU/Linux OpenSUSE 10.2,
- Postfix et Cyrus déployés sur le Dom0 afin d'assurer les besoins de la société en terme de gestion d'E-mails,
- une possibilité d'extension pour l'utilisation d'autres solutions pré-déployées dans des machines virtuelles.

En plus d'être facile à déployer, la distribution de solutions sous cette forme permet également d'avoir un environnement par défaut plus sécurisé puisque isolé des autres domaines par l'hyperviseur. Le déploiement dans Xen d'une machine virtuelle conçue à l'origine pour VMWare peut paraître étonnant, mais cela n'a en réalité rien d'exceptionnel. Une machine virtuelle est composée de deux éléments : un fichier de configuration spécifique au logiciel de virtualisation et une solution de stockage (une ou plusieurs partitions, images de disques ou images de partitions). Prenons le cas de project-open, la machine virtuelle était composée de son fichier de configuration VMWare et d'une image de disques contenant deux partitions. En écrivant un fichier de configuration équivalent pour Xen, il a été possible sans problème de démarrer une nouvelle machine complètement virtualisée sur base de cette image disque.

5.9 Aspect humain

L'utilisation de la virtualisation et de systèmes à base de terminaux légers bouleverse l'utilisation quotidienne de l'outil informatique des utilisateurs. Bien que cet aspect soit souvent éclipsé par des aspects plus techniques de l'implémentation de la virtualisation dans un environnement de travail, il faut toujours garder à l'esprit qu'une architecture informatique est avant tout un service qui devra être utilisé par des utilisateurs finaux.

- Le remplacement de la machine de bureau d'un utilisateur par un terminal léger pourrait lui laisser croire qu'il pourra faire moins de choses

avec son nouveau matériel qu'avec son ancien, certains utilisateurs pourraient même prendre ce changement d'environnement de travail comme une diminution de leur statut au sein de l'entreprise. Une information complète des utilisateurs sur les raisons qui ont poussé le service informatique de l'entreprise à faire ce choix est donc indispensable.
- Certains utilisateurs qui stockaient du contenu personnel et/ou multimédia sur leur machine de travail ne pourront probablement plus le faire en fonction de la politique de stockage centralisé choisie. Cela dit, un environnement de travail doit rester un environnement de travail et des éléments n'en faisant pas partie constituent souvent des sources de problèmes potentiels (virus, profils très lourds car utilisés pour le stockage de films ou de musique, ralentissements de la machine, ...). L'instauration d'une politique de travail accompagnée d'une information des utilisateurs sont là encore des éléments à ne pas négliger.
- Il est probable que certains appareils tels que des assistants personnels ou des appareils communicants ne puissent plus être utilisés avec certains terminaux légers. Là encore il est important d'en informer les utilisateurs finaux et d'instaurer une homogénéité au sein des outils de travail afin d'éviter tout problème de compatibilité.

Le secret de la réussite d'une bonne migration est donc avant tout basé sur une bonne information des utilisateurs car un projet informatique est avant tout un projet humain.

5.10 Les fonctionnalités de XenMonitor

XenMonitor permet la surveillance simultanée et une gestion de base de plusieurs hyperviseurs Xen distants. Il permet les fonctionnalités suivantes :

- Connexion/déconnexion d'un hôte à la demande
- Option de connexion automatique à un hôte au démarrage du logiciel
- Option de reconnexion automatique en cas d'échec ou de perte de connexion
- Gestion des fonctionnalités de base des machines virtuelles : démarrage, arrêt, redémarrage et arrêt brutal
- Gestion de la migration en direct d'une machine virtuelle
- Surveillance graphique de l'utilisation des VCPUs des domaines
- Consultation de données techniques concernant les domaines
- Historique des opérations effectuées au sein d'un journal
- Gestion des configurations d'hôte simple et intuitive
- Gestion des préférences utilisateur
- Gestion multilingue

5.11 La problématique de la gestion des machines complètement virtualisées

La virtualisation complète sous Xen a des inconvénients, tels que les pertes de performances en calcul ou au niveau des périphériques ou encore l'impossibilité d'allouer plusieurs processeurs virtuels à une machine virtuelle. Cependant, un autre problème impactant directement l'utilisation de XenMonitor est également d'actualité avec la virtualisation complète sous Xen : la gestion d'un arrêt « propre »de la machine virtuelle.

L'API Xen présente deux manières d'arrêter une machine virtuelle :
« clean_shutdown »(utilisé par la commande xm shutdown) et
« hard_shutdown »(utilisé par la commande xm destroy). Deux fonctionnalités que j'ai implémenté dans XenMonitor comme les fonctions « Arrêter une VM »et « Tuer une VM ». La première permet d'arrêter la machine virtuelle en terminant correctement tous les processus du système invité et sans risquer de perdre des données ou de corrompre le système de fichiers de l'image disque du système invité. La deuxième est à réserver aux cas de plantages ou de mauvais fonctionnement du système invité et aboutit à un arrêt brutal de la machine virtuelle, avec les risques que cela comporte. Ces deux comportements peuvent respectivement être comparés à l'arrêt d'une machine suite à la demande de l'utilisateur ou suite à une coupure de courant.

Sous Xen, dans une machine complètement virtualisée, l'arrêt propre n'est pas supporté et l'appel à cette fonction n'effectue en fin de compte qu'un arrêt brutal de la machine virtuelle. Ce problème étant le plus souvent rencontré avec Windows, puisqu'il ne peut être paravirtualisé, une solution a été envisagée afin de pallier à ce problème. Le projet s'appelle WindowsXenPV et est disponible à l'adresse http ://wiki.xensource.com/xenwiki/XenWindowsGplPv

WindowsXenPV est constitué de drivers et d'un démon chargé d'intercepter les demandes d'arrêt dans le but d'effectuer une demande au système afin qu'il s'éteigne par lui-même, corrigeant ainsi le problème d'arrêt brutal constaté sur les machines complètement virtualisées et fonctionnant sous Windows. Il est à noter que Virtualbox, un logiciel de virtualisation ne faisant que de la virtualisation complète, offre une option alternative à l'extinction d'une machine virtuelle : l'envoi d'une demande d'extinction ACPI. Au moment de la demande, le système virtualisé croit donc que l'utilisateur a poussé sur le bouton d'arrêt de sa machine, entraînant ainsi une extinction propre de la machine virtuelle. Dommage que l'équipe de développement de Xen

n'ait pas implémenté une fonctionnalité similaire.

WindowsXenPV permet également, grâce aux drivers fournis, d'exploiter les périphériques de bloc et réseau du Dom0 comme le ferait un environnement paravirtualisé. Bien que des gains de performances aient été reportés par des utilisateurs, la solution reste encore pour le moment expérimentale, son évolution est cependant assez rapide.

5.12 Cas concret d'utilisation de XenMonitor

La société Unipress a pris contact avec plusieurs sociétés intéressées par l'utilisation de XenMonitor, dont la société Vandenborre. Ceux-ci sont prêts à tester l'application et à donner un avis d'utilisateur final sur le produit. Cependant, utilisant actuellement la solution de virtualisation VMWare et n'ayant pas encore commencé leur migration vers Xen, je ne peux malheureusement pas inclure leurs commentaires dans le présent document.

6 Analyse

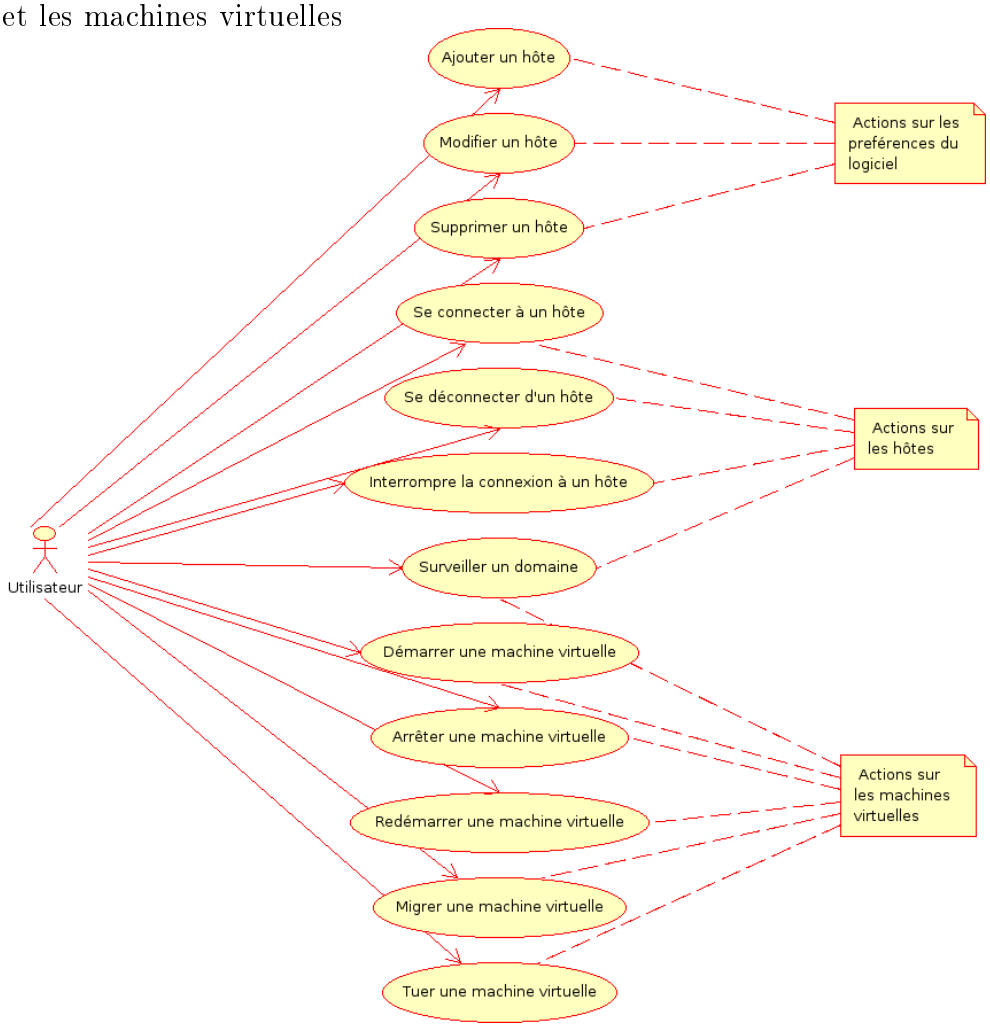
6.1 Cycle de développement

Beaucoup de libertés m'ont été laissées dans la réalisation du projet, telles le choix du langage de développement ou encore des outils, mais également dans la présentation et les fonctionnalités du logiciel lui-même, seuls quelques critères à respecter m'ont été fournis au départ. Afin de ne pas m'écarter de ce que mon promoteur attendait comme résultat du développement, j'ai choisi d'utiliser un cycle de développement itératif, me permettant ainsi régulièrement d'avoir un retour sur le travail effectué jusqu'à présent, de connaître les éventuelles modifications à effectuer sur l'existant et les nouvelles fonctionnalités à implémenter par la suite.

6.2 Actions sur les hôtes et les machines virtuelles

Voici un diagramme de cas d'utilisations présentant les actions possibles sur les hôtes ainsi que les machines virtuelles. La récupération de données n'y figure pas, étant une action récurrente effectuée par le logiciel une fois la connection correctement établie avec l'hôte.

FIG. 2 – Diagramme de cas d'utilisations des actions permises sur les hôtes et les machines virtuelles



6.3 Choix du langage

Plusieurs associations de technologies permettaient d'atteindre les critères de base recherchés :

- Java + portage de l'API Xen XML-RPC + Swing
- Perl + module RPC : :XML + PerlQt
- Perl + module RPC : :XML + wxPerl
- Python + API Xen XML-RPC + PyGtk
- Python + API Xen XML-RPC + PyQt
- Python + API Xen XML-RPC + Tkinter
- Python + API Xen XML-RPC + wxPython

Java pouvait tout-à-fait correspondre aux critères recherchés, Swing est très complet et permet de créer des interfaces graphiques très intuitives. Cependant, en comparant l'API Xen XML-RPC, écrite à la base en Python, avec son homologue porté en Java, on se rend vite compte de l'efficacité de Python en terme d'efficacité par rapport au nombre de lignes de code.

Perl, quant à lui, ne possèdait pas (encore) de portage de l'API Xen XML-RPC au moment de mon choix, ce qui obligeait l'utilisation du module RPC : :XML afin d'en écrire une. De plus, l'utilisation de Perl ne présentait pas d'avantage particulier.

Python est le langage dans lequel la totalité des scripts fournis avec Xen ont été écrits, y compris l'API Xen utilisée par ces scripts. Python est un langage de programmation interprété à typage dynamique fort et favorisant la programmation orientée objet. Apparu en 1990 et bien qu'ayant chez bon nombre de programmeurs une sombre réputation de langage dépassé, son utilisation est en pleine expansion. Il est utilisé dans de nombreux nouveaux projets sous GNU/Linux, il est également utilisé par la NASA notamment afin de calibrer et d'analyser les données recueillies par le téléscope spatial Hubble. Des nombreuses firmes telles que AstraZeneca et Honeywell l'utilisent également dans certains projets. Python a été placé sous une licence libre proche de la licence BSD.

Tkinter permet d'obtenir très rapidement et facilement des interfaces utilisateur graphiques, de plus il est fourni par défaut avec Python. Cependant il ne permet malheureusement de faire que des interfaces graphiques simplistes. wxPython est l'alternative à Tkinter la plus utilisée, mais ne présente comme Tkinter que la possibilité de créer des interfaces graphiques. PyGtk aurait

lui aussi pu être un bon candidat, mais ne présente pas non plus de fonctionnalités autre que la création d'interfaces graphiques.

Qt 4 présente un rendu graphique assez séduisant et des possibilités d'utilisation de la programmation MVC, mais il présente également un framework complet offrant des possibilités d'utilisation très variées. De plus, Qt a fait ses preuves dans un projet assez conséquent : le bureau KDE, qui constitue d'ailleurs en partie mon environnement de travail. Il est également utilisé dans le monde de l'entreprise par de grandes sociétés telles que la NASA, Google ou encore Adobe Systems. Mais il est également présent dans le monde de l'embarqué, grâce à Qtopia, une plate-forme applicative libre destinée aux appareils mobiles sous GNU/Linux. Qt 4 est disponible sous triple licence Trolltech/GPLv2/GPLv3, ce qui permet de profiter de Qt sous licence GPL à condition d'en respecter les termes. Dans le cas contraire, les projets à buts commerciaux désirant utiliser Qt devront s'acquitter de droits d'utilisation de Qt.

Bien que Python et Qt soient tous les deux des technologies mûres et ayant déjà fait leurs preuves, leur utilisation commune (grâce à l'interface de programmation PyQt) n'est pas très répandue. Cependant, de nombreux projets récents sous GNU/Linux utilisent cette technologie, notamment au sein du bureau KDE. Bien qu'existant depuis longtemps, on peut donc qualifier cette association de valeur montante.

Je mentirais si je disais que mon choix n'a pas été influencé par le fait que je n'avais jamais approché ces deux technologies auparavant, mais il va de soi que mon envie d'apprendre n'était pas prioritaire par rapport à un choix judicieux et adapté des technologies utilisées afin de mener à bien ce projet.

6.4 L'API Xen XML-RPC

L'API Xen XML-RPC est une interface permettant de contrôler Xen par l'intermédiaire du démon de contrôle de Xen. La communication se fait au moyen de requêtes HTTP et de messages XML. Cette API est relativement récente et est apparue avec la version 3.0.4, en effet, une version plus ancienne et maintenant dépréciée de l'API XML-RPC existait auparavant. Celle-ci avait la réputation d'être peu stable et un des buts premiers de la nouvelle API XML-RPC était de remédier à ce problème tout en maintenant temporairement le support de l'ancienne API. Entre-temps, le projet libvirt est né, fournissant une interface commune à la gestion de Xen, QEMU, KVM,

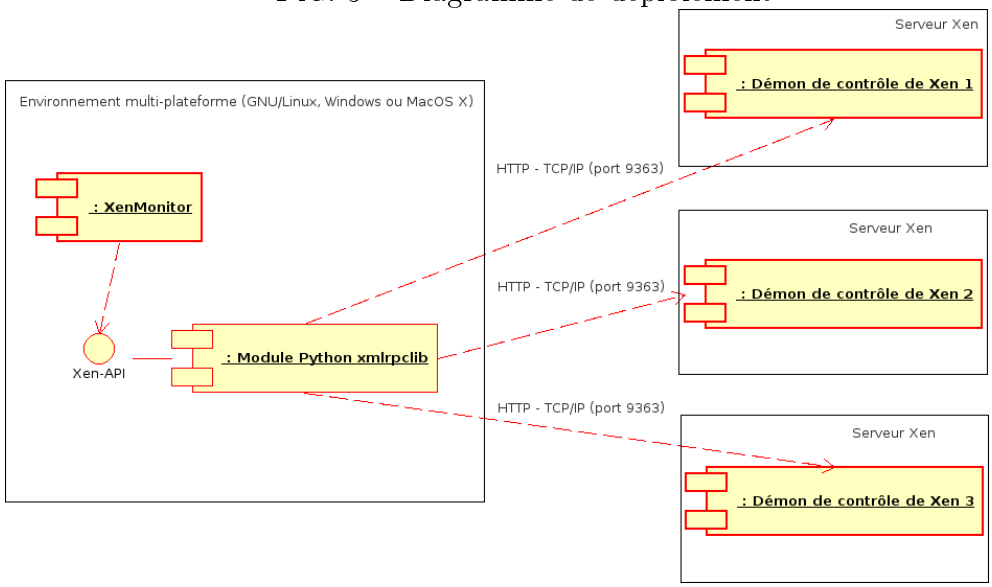
LXC et OpenVZ. Cependant, à ce jour, la gestion de Xen par la libvirt s'effectue toujours au travers de l'ancienne API Xen XML-RPC. Les besoins d'Unipress en matière de virtualisation étant strictement limités à Xen et l'utilisation de la libvirt représentant une couche d'abstraction supplémentaire et inutile dans ce cas-ci, mon choix s'est finalement porté, avec l'accord de mon promoteur de stage, sur la nouvelle API Xen XML-RPC. Cependant, si dans des développements futurs le besoin d'utiliser la libvirt se faisait sentir, l'architecture du logiciel est assez modulaire que pour pouvoir intégrer son utilisation (éventuellement conjointe à l'API Xen XML-RPC) au sein du logiciel sans devoir repenser tout son fonctionnement.

6.5 Déploiement

XenMonitor étant écrit en Python, il peut donc être exécuté dans tous les environnements supportés par l'interpréteur.

Une fois lancé, XenMonitor se connecte aux différents hôtes Xen, plus précisément au démon de contrôle de Xen de chaque hôte, afin d'y récupérer les données à contrôler. Cette communication s'effectue au travers de requêtes XML-RPC, grâce à l'API Xen XML-RPC, elle-même exploitant le module xmlrpclib de Python.

FIG. 3 – Diagramme de déploiement



La communication avec le démon de contrôle de Xen par l'intermédiaire de l'API Xen XML-RPC s'effectue en 3 temps :

Identification/authentification : L'appel à *session.xenapi.login_with_password()* permet de s'identifier et de s'authentifier auprès du démon de contrôle de Xen, il en résulte un identificateur de session qui sera utilisé dans les requêtes suivantes.

Récupération des données : Des requêtes peuvent maintenant être effectuées afin de récupérer des informations.

Déconnexion : L'appel à *session.logout()* permet d'invalider la session courante.

Une alternative séduisante à l'utilisation d'une interface graphique directement sur la machine client aurait été d'implémenter l'application sous forme d'interface web. Cependant, ce choix est loin d'être idéal. En effet, une application web nécessite un serveur web pour fonctionner et le rendre accessible à des clients. Cependant, un serveur web nécessite une machine constamment en ligne afin d'assurer un service satisfaisant aux clients, ce dont une petite entreprise qui a opté pour la virtualisation ne dispose pas forcément. Plusieurs solutions s'offrent alors à nous :

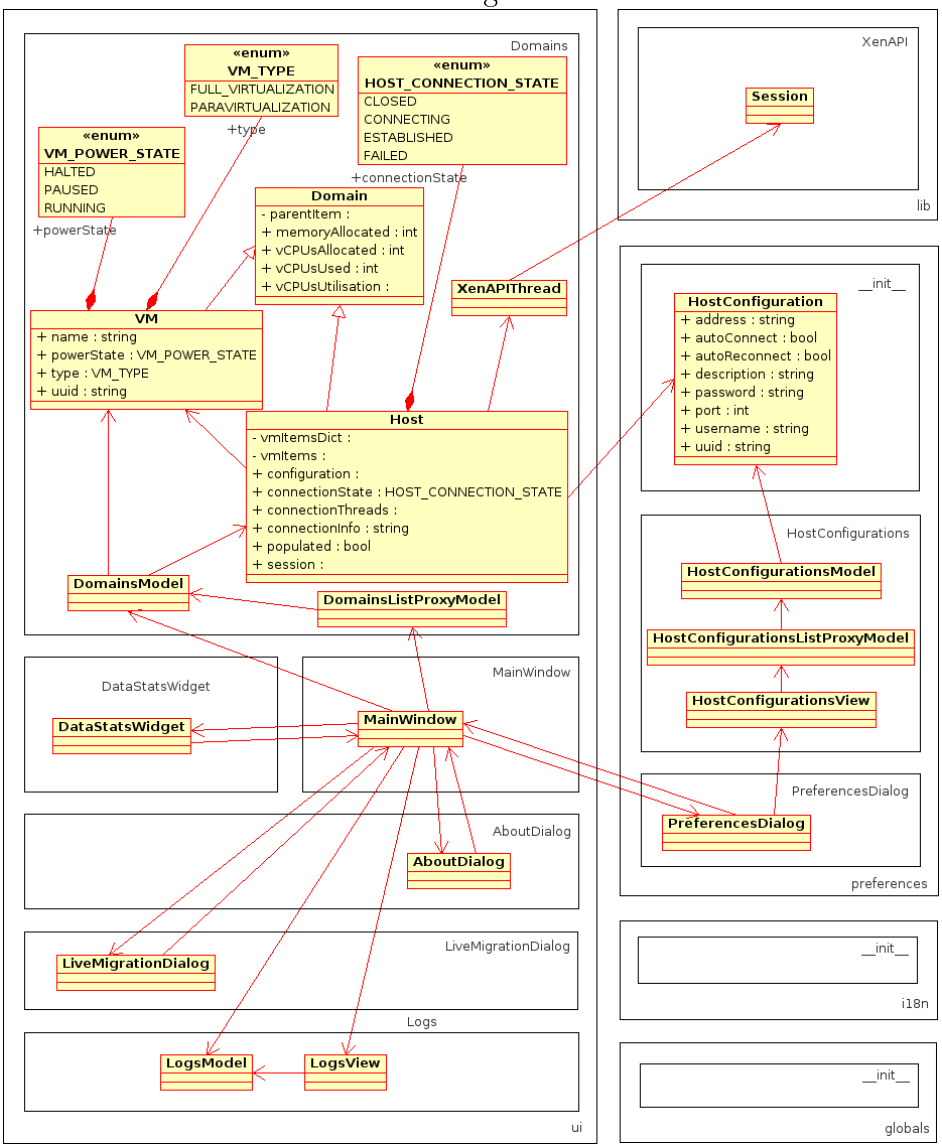
- Utiliser une machine client afin d'héberger le serveur web est probablement la pire solution. Une machine client ne dispose en général pas d'une adresse IP fixe ou d'une adresse DNS la rendant facilement accessible aux autres machines clientes. De plus, de par sa nature de machine cliente, elle ne se doit pas d'être constamment disponible et a plus de probabilité de tomber en panne qu'un serveur par exemple.
- Utiliser le Dom0 d'une machine virtualisante aurait été envisageable, mais si cette machine devient indisponible, comment continuer à surveiller les autres ? Et sur quelle machine installer l'application ?
- Utiliser un DomU d'une machine virtualisante présente les mêmes inconvénients que le Dom0, en plus du fait que le système de surveillance serait indisponible une fois la machine virtuelle assurant son fonctionnement stoppée.
- Utiliser un collecteur de données sur chaque machine virtualisante afin de récupérer les informations en temps voulu sur une machine cliente. C'est une approche intéressante et qui n'est pas incompatible avec l'architecture actuelle. Elle mérite réflexion et fait d'ailleurs partie des améliorations envisageables pour le futur de l'application.

On se rend vite compte que si l'on ne dédiait pas un serveur fiable pour une application orientée web, elle constituerait un « single point of failure ». Au contraire, une architecture de logiciel client se connectant directement aux machines virtualisantes permet de vérifier l'accessibilité et l'état de chacune d'elle sans risquer de perturber l'accès aux informations des autres machines virtualisantes. Cette option permet d'ailleurs également, si le besoin s'en fait sentir, d'utiliser le logiciel sur le Dom0 ou un DomU d'une machine virtualisante.

6.6 Architecture du logiciel

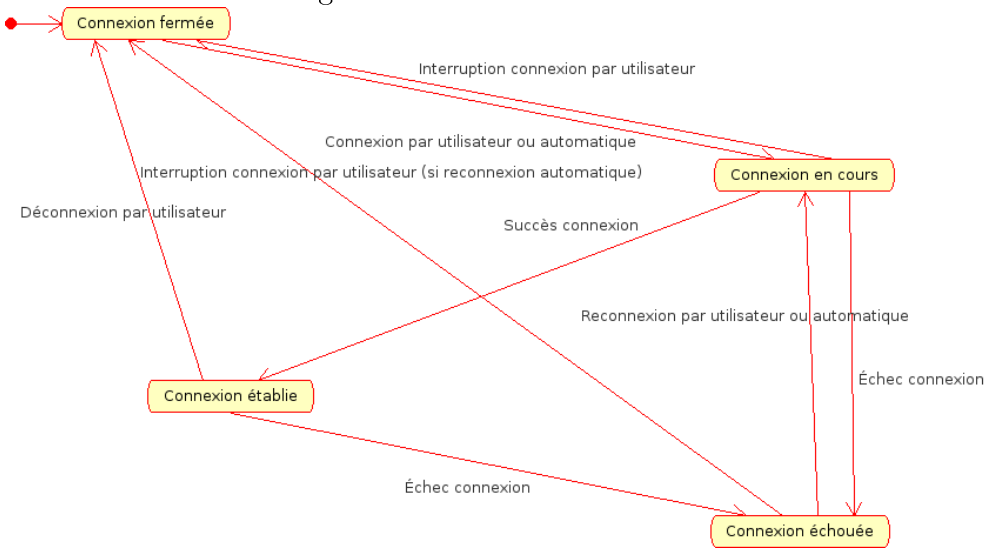
Voici un diagramme de classes présentant l'architecture du logiciel dans son ensemble. Pour des raisons évidentes de lisibilité et afin de rendre la compréhension plus rapide, j'ai volontairement omis d'indiquer les classes du framework Qt utilisées telles quelles ainsi que les relations les impliquant dans l'architecture du logiciel. J'ai également limité le détail des attributs aux classes impliquées dans la logique même de l'application.

FIG. 4 – Diagramme de classes



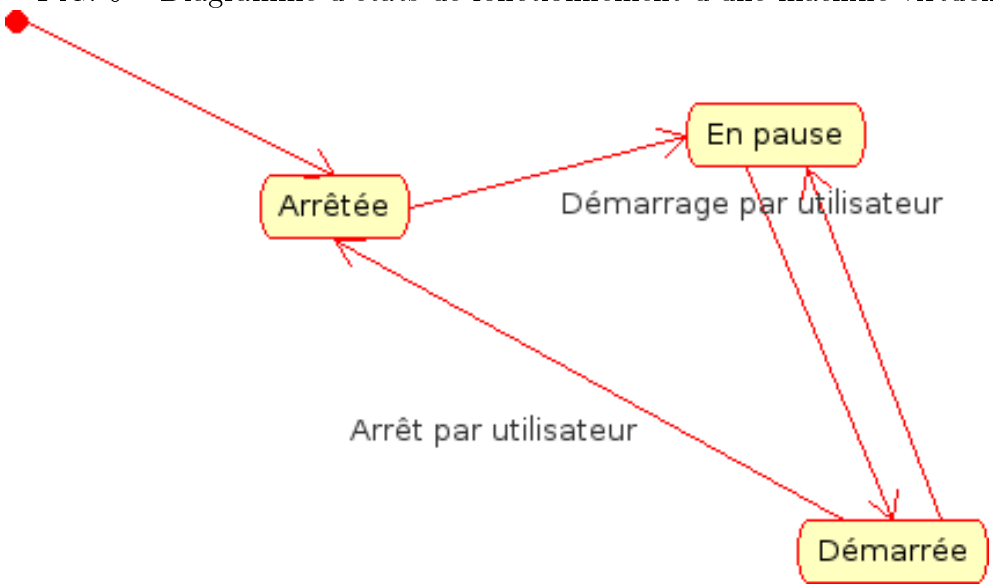
Il peut être intéressant d'attirer l'attention sur les différents états que peuvent prendre les hôtes (état de connexion) et les machines virtuelles (état de fonctionnement).

FIG. 5 – Diagramme d'états d'une connexion à un hôte



Les états présentés à la figure 5 n'existent qu'au sein du logiciel lui-même, le notion de connexion à un hôte représentant en réalité l'état de la communication avec le serveur en tant que machine physique.

FIG. 6 – Diagramme d'états de fonctionnement d'une machine virtuelle

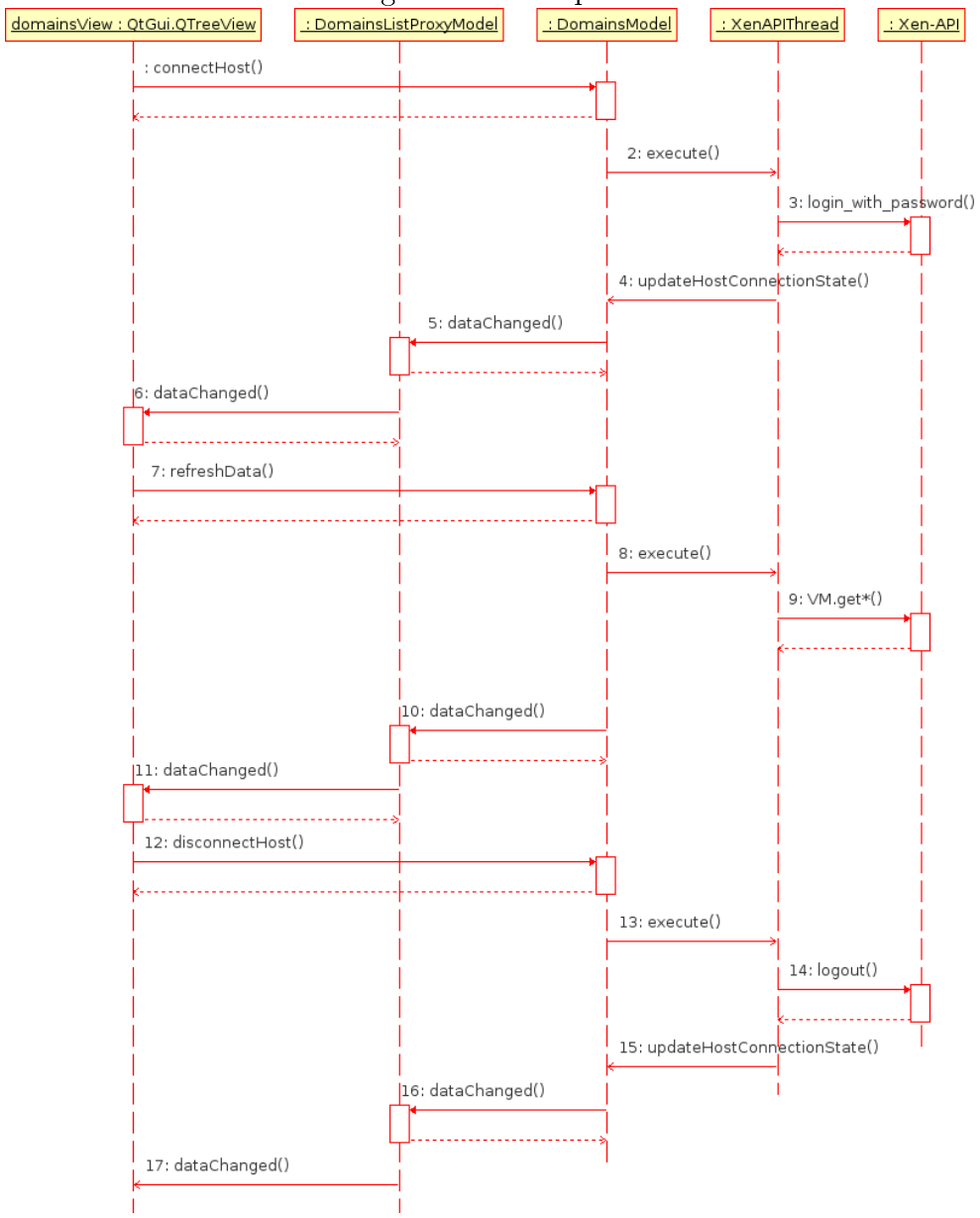


Les états présentés à la figure 6 existent au sein du logiciel, mais également au sein de Xen. Il s'agit de l'état de fonctionnement des machines virtuelles, qui peut être modifié par l'intermédiaire du logiciel, mais également par d'autres logiciels utilisant l'API Xen, tels que la commande xm par exemple. La notion d'état de pause est ainsi réprésentée au sein du logiciel mais il n'existe pas de commande permettant de mettre la machine dans cet état (celle-ci sera ajoutée dans le futur si elle est nécessaire), elle peut par contre l'être par l'intermédiaire de la commande xm. De même, une machine virtuelle arrêtée passe d'abord par un état de pause avant d'être démarrée, bien qu'aucune demande explicite de mise en état de pause n'ait été effectuée.

6.7 Séquences des actions

Voici un diagramme de séquences présentant une connexion à un hôte, un rafraîchissement des données ainsi qu'une déconnexion de l'hôte.

FIG. 7 – Diagramme de séquences des actions



7 Qt et ses facilités

N'ayant pas une vue globale sur les possibilités que m'offrait Qt, il m'est arrivé durant le développement de XenMonitor d'implémenter des fonctionnalités implémentées dans le framework et donc facilement utilisables. Cela dit, il n'y a pas de meilleur apprentissage que celui que l'on tire de ses erreurs. J'ai donc réimplémenté ces fonctionnalités en utilisant les outils que Qt mettait à ma disposition. Les raisons pour lesquelles j'ai décidé de réimplémenter en utilisant Qt des fonctionnalités que j'avais déjà écrites moi-même sont multiples :

- Cela m'a permis d'acquérir une expérience dans l'utilisation de Qt, tout en ayant un regard sur la manière dont j'aurais pu implémenter ces fonctionnalités.
- L'utilisation d'un framework permet de ne pas « réinventer la roue »et d'écrire moins de lignes de code. Utiliser un framework bien conçu permet de limiter les risques d'erreurs et de comportements indésirables en permettant au programmeur de se concentrer sur la logique de la programmation plutôt que sur des tâches routinières et répétitives.
- Les éléments du framework Qt suivent une certaine logique de fonctionnement, les rendant interopérables entre eux. Écrire du code en utilisant un framework permet d'écrire des fonctionnalités plus modulaires et plus homogènes.
- Le code étant plus modulaire, la maintenance en est grandement facilitée.

Pour ces raisons, je vous propose d'aborder quelques facilités de Qt qui m'ont été utiles lors du développement de XenMonitor.

7.1 La programmation MVC

Qt permet deux approches en ce qui concerne les éléments graphiques. La première consiste à utiliser des QWidgets, des éléments graphiques possédant un modèle intégré. Ceux-ci présentent l'avantage d'être très rapides à prendre en main, ils utilisent le pattern MVC au sein de leur implémentation mais ne permettent pas au programmeur de gérer lui-même son modèle de données. Les QWidgets conviendront aux petits projets, aux fonctionnalités simples n'ayant pas l'ambition d'évoluer dans le futur et aux programmeurs pressés ou peu soucieux de la maintenabilité de leur logiciel.

La deuxième approche consiste à utiliser des Q*Views et des Q*Models, permettant ainsi au programmeur, au prix d'un temps d'apprentissage et d'un temps de programmation légèrement plus long, de gérer lui-même un modèle

de données dissocié de la vue. Qt fournit plusieurs modèles de données : un modèle de données standard permettant de créer rapidement un modèle de données simple, deux modèles de données abstraits adaptés aux listes et aux tableaux, et un modèle abstrait autorisant des modèles de données plus complexes ou hiérarchiques.

Ma première approche dans la programmation de la gestion des domaines de XenMonitor (l'affichage principal) fut d'utiliser un QTreeWidget. Face à la complexité du problème et de la conception d'un modèle hiérarchique personnalisé et suite à mon manque d'expérience dans la gestion de modèles avec Qt, j'ai repoussé l'utilisation d'un modèle. Cependant, les limites du QTreeWidget se sont vite fait sentir tant au niveau de l'optimisation du code que de la maintenabilité de celui-ci, j'ai donc décidé d'utiliser un modèle centralisé pour la gestion des domaines, me permettant ainsi s'y connecter la vue principale, la vue des propriétés du domaine sélectionné ainsi que la liste présentée par la boîte de dialogue de migration d'une machine virtuelle.

7.2 Les modèles et leur place dans Qt

Les modèles ont une place toute particulière dans Qt, ils ne se contentent en général pas de contenir passivement les données qu'on leur confie. Bien souvent, c'est eux-même qui se chargent de contacter la ou les source(s) de données, que ce soit une base de données grâce au modèle QSqlTableModel fourni en standard par Qt, ou des sources diverses telles que des préférences utilisateur et des réponses à des requêtes XML-RPC, comme dans le cas du modèle de gestion des domaines implémenté dans XenMonitor. Ce rôle donné au modèle permet une implémentation aisée de l'exportation des données d'un modèle dans un fichier texte par exemple, comme c'est le cas avec le journal de XenMonitor, dont la fonctionnalité d'exportation tient en quelques lignes.

7.3 Les modèles proxy

Les modèles de données de Qt utilisant une interface similaire, l'affichage des données peut être effectué sur n'importe quel type de vue, sans nécessiter de modifier celui-ci. Seule la vue hiérarchique nécessite une extension de l'interface standard afin de prendre en compte le concept de parenté des éléments. Mais lors de l'utilisation de multiples vues avec un modèle unique, l'intérêt aurait été très limité si l'affichage des données était identique sur chaque vue. C'est là qu'intervient le concept de modèle proxy. Un modèle proxy est un modèle qui, comme son nom l'indique, implémente le pattern

proxy. Il utilise un modèle comme source de données et, en utilisant l'interface standard des modèles Qt, présente ces données en fonctions de certains critères. Deux modèles proxy sont fournis avec Qt, un modèle proxy intégrant des possibilités de tri et de filtrage, et un modèle proxy abstrait permettant au programmeur d'implémenter un modèle proxy sans fonctionnalité de tri ou de filtrage. C'est ainsi que la vue principale de XenMonitor présente une vue d'ensemble des domaines pouvant être triée selon les souhaits de l'utilisateur, tandis que la vue des propriétés du domaine sélectionné affiche les détails du domaine sélectionné. C'est également grâce à ce concept que la gestion des configurations d'hôte dans la boîte de dialogue des préférences permet le filtrage des éléments et leur tri.

7.4 Le système de signaux et de slots

La communication entre les différents éléments du framework Qt se fait principalement au moyen de signaux et de slots. Prenons par exemple la cas d'un modèle et d'une vue, on va signaler à la vue quel modèle contient les données que celle-ci devra afficher. Lorsqu'une données est modifiée dans le modèle, celui-ci doit prévenir la vue afin qu'elle puisse récupérer la nouvelle donnée et la présenter à l'utilisateur. Cette opération s'effectue au moyen d'un signal au niveau du modèle qui sera émis au moment où une donnée est modifiée. L'opération qui consiste à lier le modèle à la vue aura au préalable connecté ce signal à un slot de la vue chargé de traiter la modification de la donnée et de faire le nécessaire afin d'actualiser l'affichage. Ainsi, lors de l'émission du signal *dataChanged()* par le modèle, la vue sera informée de la modification d'une donnée et pourra ainsi actualiser son affichage.

7.5 Le mapping entre modèle et widgets

Il peut être intéressant d'effectuer un mapping entre des QWidgets simples, tels que des QLineEdits (un élément permettant la saisie de texte sur une ligne), Qt prévoit une solution pour ce cas de figure : le QDataWidgetMapper. En le connectant à un signal de sélection d'une vue par exemple, on peut facilement obtenir l'affichage du contenu de la ligne du modèle sélectionnée dans des QWidgets. Le QDataWidgets gère également la modification des données et permet ainsi de sauvegarder le contenu du QWidget modifié à la demande, ou à la perte du focus sur celui-ci et d'ainsi mettre-à-jour le contenu du modèle. C'est ainsi que la gestion des configurations d'hôte permet la modification dynamique des données de ses éléments au travers d'un formulaire.

7.6 L'internationalisation

Qt apporte un réel confort en ce qui concerne la gestion de l'internationalisation. Le programmeur n'a qu'à utiliser la méthode *tr()* implémentée par toutes les classes du framework au moment de la rédaction de chaque chaine de caractère à traduire. Il devra ensuite prévoir une gestion plus ou moins poussée des langues, s'il souhaite que son application utilise simplement la langue du système sur lequel il devra fonctionner ou s'il souhaite laisser ce choix à l'utilisateur final. Dans le cas de XenMonitor, c'est ce dernier choix que j'ai implémenté, puisque XenMonitor permet de choisir une langue parmi celles disponibles dans la boîte de dialogue des préférences. Une fois cela implémenté, il ne reste plus qu'à générer des fichiers de traduction qui permettront de faire le lien entre les chaînes de caractères originales et leur traduction dans une langue donnée. La traduction en elle-même est une étape qui peut être totalement dissociée et confiée à un traducteur sans expérience spécifique en informatique, le logiciel Qt Linguist lui permettra d'effectuer les traductions dans un environnement simple et intuitif.

7.7 Les ressources

Une application est constituée de code source, mais également souvent de ressources comme des icônes, des fichiers de traduction, des pages HTML, ... Ces ressources peuvent poser deux principaux problèmes :

- leur chemin d'accès relatif est différent selon le fichier source qui les utilise, contraignant le programmeur à tenir compte de la place du fichier source dans l'arborescence du logiciel,
- leur distribution peut être difficile du fait de leur nombre parfois important.

Qt propose une solution à ces deux problèmes : les QResources. Elles se présentent sous forme d'un fichier XML référençant toutes les ressources utilisées par le logiciel puis, après une phase de compilation, ne consistent plus qu'en un unique fichier binaire. Ce fichier binaire peut alors être accédé de partout dans le code du logiciel et présente une arborescence interne identique peu importe l'endroit où on l'utilise. Il va de soi que pour déployer l'application, la copie d'un seul et unique fichier est plus simple que la copie de plusieurs répertoires contenant des dizaines voire des centaines de fichiers. Ce système a été utilisé pour la gestion des icônes utilisés par XenMonitor, mais pas pour ses fichiers de traduction. La raison de ce choix est la suivante : toutes les installations n'ont pas forcément besoin de toutes les langues. Ce sont des fichiers qui doivent respecter une certaine modularité, l'ajout du support d'une langue représentant, dans le cas d'une utilisation d'une QResource, le

remplacement total de toutes les ressources. La plupart des systèmes d'empaquetage sous GNU/Linux incite également à séparer le support des langues de l'application pour ces mêmes raisons.

7.8 La gestion des préférences

La gestion des préférences et leur persistance pourrait être implémentée grâce à un fichier XML stocké dans un dossier utilisateur par exemple, c'était d'ailleurs ma première approche pour XenMonitor. Cependant, Qt met à disposition des programmeurs une solution qui permet une persistance spécifique en fonction du système d'exploitation tout en présentant une interface de programmation identique. Il s'agit des QSettings, ceux-ci permettent la persistance de données simples comme de données dont le nombre est indéfini (une liste ou un tableau par exemple), ils permettent également de structurer ces données de manière hiérarchique au moyen de groupes. La lecture et l'écriture se fait au moyen d'une clé associative et des méthodes *value()* et *setValue()*. Il devient ainsi aisément implémentable de stocker des préférences utilisateur ou encore de sauvegarder l'état actuel d'un widget dans le but de les restaurer à la prochaine utilisation du logiciel. Les QSettings utilisent donc des méthodes de persistance différentes en fonction du système d'exploitation :

- sous GNU/Linux, c'est un fichier texte qui sera écrit dans un sous-dossier de configuration du dossier personnel de l'utilisateur,
- sous Mac OS, un fichier XML sera utilisé,
- tandis que sous Windows, c'est dans la base des registres que les préférences seront stockées.

7.9 L'assistance à la création d'interfaces

Qt propose un outil, nommé Qt Designer, permettant la création d'interfaces graphiques simples, de manière graphique et intuitive. L'outil, permettant de dessiner rapidement des écrans, permet au programmeur et à l'utilisateur final d'avoir endéans un laps de temps très court, une idée du rendu graphique de l'application. Qt Designer permet donc de pouvoir dessiner des écrans réels au moment de l'analyse, mais également de pouvoir les modifier très facilement en fonction de besoins, et de générer du code afin d'utiliser l'interface au sein de l'application. Beaucoup d'options sont déjà disponibles afin de paramétrer les éléments graphiques des interfaces, mais on regrettera qu'il n'y en ait pas davantage. La tendance actuelle est à la génération d'interfaces et ce genre de solution ne pourra qu'être plus indispensable encore dans les prochaines versions.

7.10 La documentation

La documentation officielle Qt est disponible à l'adresse
http ://doc.trolltech.com, elle est cependant également disponible depuis Qt Assistant, un logiciel également fourni par Trolltech. Celui-ci permet donc la consultation hors-ligne de la documentation disponible en ligne, mais également des recherches avancées dans le contenu de celle-ci, la gestion de marque-pages ou encore la recherche via un index.

7.11 La librairie Qwt

La librairie Qwt est une librairie de dessin de graphiques. Bien qu'elle ne fasse pas partie intégrante de Qt, elle s'appuie fortement sur ces concepts et constitue une référence en matière de dessin de graphiques avec Qt. Les dock widgets de surveillance implémentés dans XenMonitor exploitent les possibilités offertes par cette librairie. La classe QwtPlot utilisée à cet effet ne permettant pas l'association avec un modèle de données, le timer intégré de celle-ci a été utilisé afin d'actualiser les données de manière récurrente. On pourrait citer Matplotlib comme concurrent de Qwt, mais il semblerait qu'il s'utilise plus difficilement de manière conjointe à Qt.

8 Conclusion

8.1 Objectifs atteints

Les fonctionnalités du XenMonitor figurant dans la liste des critères cités en début de document sont toutes opérationnelles. Bien que le projet puisse encore être amélioré de bien des manières, ce qui a été accompli constitue une bonne base pour une application qui évoluera au fil du temps.

8.2 Problèmes rencontrés

Parmi les problèmes que j'ai rencontré durant mon stage, on pourrait citer un manque de vision globale sur les technologies qui étaient à ma disposition. Cette remarque concerne particulièrement Qt. Ne disposant pas de développeur Qt dans mon entourage, cette vision globale des possibilités ne pouvait s'obtenir que par une expérience acquise durant la phase de développement même du logiciel, occasionnant ainsi des retards et des réimplémentations de certaines parties du logiciel.

Le manque d'exemples complexes concrets pouvant s'appliquer à XenMonitor dans les technologies de développement choisies aurait également pu être comblé par une expérience dans le domaine. Un laps de temps supplémentaire a donc été nécessaire afin de découvrir certains concepts par moi-même.

Concernant les problèmes techniques n'étant pas directement liés au développement du logiciel, on pourrait citer le bug survenu sous OpenSUSE 10.3 et rendant impossible le lancement de Virt-manager, l'outil de gestion pour Xen fourni en standard dans cette distribution, une fois les logiciels de la machine mis à jour (voir semaine 7 du calendrier en 4.7). Ce problème a également été reporté sur des systèmes en production, pour lesquels celui-ci a pu être corrigé suite à mon analyse.

8.3 Ce que le stage m'a apporté

Ce stage m'a permis de me familiariser avec bon nombre de nouvelles technologies telles que la virtualisation avec Xen ou encore la programmation en Python avec Qt, mais ce que j'ai appris durant cette période ne s'arrête pas là. J'ai eu l'occasion de déployer et d'expérimenter ces technologies sur du matériel performant, digne de machines destinées à la virtualisation en environnement de production. J'ai également pu me familiariser avec du matériel et de nouveaux constructeurs du monde des serveurs dont je connaissais à

peine l'exxistence. Cela dit, le stage ne m'a pas apporté qu'un apprentissage purement technique, il m'a aussi donné un aperçu de la vie en tant qu'employé dans une entreprise tant sur le point professionnel que relationnel, de la manière dont s'organise une entreprise et des difficultés que peuvent rencontrer les responsables de celle-ci face à un marché en constante évolution.

8.4 Améliorations possibles de XenMonitor

Un logiciel n'étant jamais terminé, voici quelques fonctionnalités qu'il pourrait être intéressant de développer dans le futur :

Système d'alarme Un système d'alarme permettant d'être averti par E-mail ou SMS, parallèlement au journal déjà présent dans l'application. De même, ce système pourrait être étendu à la surveillance de l'utilisation des VCPUs et permettre une personnalisation du niveau d'alerte par l'utilisateur.

Migration de machine virtuelle par « glisser-déplacer » Une utilisation du « glisser-déplacer »pourrait être plus intuitive qu'une boîte de dialogue pour la migration de machines virtuelles, à confirmer par les utilisateurs finaux.

Sécurité Le support des connexions SSL avec les hôtes Xen afin de sécuriser les communications au sein d'un environnement de production.

Accès VNC Xen mettant à disposition, s'il est configuré pour, un accès aux machines virtuelles par VNC, il pourrait être intéressant d'intégrer un client VNC au sein de l'application.

Gestion d'un espace de stockage commun des machines virtuelles L'API Xen ne permettant pas d'effectuer d'action sur les machines virtuelles dont la configuration n'a pas été chargée, il pourrait être utile de pouvoir gérer un ensemble de configurations de machines virtuelles dans un espace de stockage commun.

Possibilité de modifications des propriétés des machines virtuelles Nom des machines virtuelles, nombre de processeurs alloués, mémoire allouée, ...

Console Accès à la console délivrée par Xen et permettant une interaction directe avec le système virtualisé en mode console.

Statistiques de l'utilisation des VCPUs En complément des graphiques de surveillance de l'utilisation des VCPUs, des statistiques sur ces informations pourraient être utiles (moyenne, minimum et maximum sur la dernière heure).

Module collecteur de données XenMonitor étant censé fonctionner sur une machine client, cette même machine n'est pas destinée à fonctionner en permanence. Il pourrait donc être intéressant de développer un démon qui fonctionnerait sur le Dom0 de la machine virtualisante et qui se chargerait de collecter en permanence des informations sur celle-ci et ses machines virtuelles. Les informations collectées pourraient ensuite être récupérées dans XenMonitor.

8.5 Critique de Python et Qt

L'utilisation conjointe du langage Python et du framework Qt permet une grande efficacité en terme de développement graphique rapide. Et bien que le framework Qt soit à la base conçu pour une utilisation en C++, l'interface de programmation PyQt est dans l'ensemble plutôt bien réussie. Cependant, un manque cruel de ressources documentaires est à combler. Bien que PyQt présente une interface similaire (à quelques détails près) à la version originale de Qt pour C++, rendant ainsi utilisable la documentation distribuée par Trolltech, l'utilisation de cette documentation destinée à des développeurs C++ peut perturber au premier abord. Un temps d'adaptation peut être nécessaire. Et bien que cette documentation soit agrémentée de quelques bouts de codes, on est souvent laissé sur sa faim... Trolltech distribue également des exemples d'utilisation de Qt, également portés en Python, mais dont on perçoit rapidement les limites lors d'une utilisation un peu plus poussée du framework. Les ouvrages sur le sujet, bien que de qualité, se comptent sur les doigts de la main, et sont inexistants dans la littérature francophone. Là encore il faudra se référer à des ouvrages traitant de Qt et de C++. Une aide peut cependant être trouvée auprès de la communauté, sur des forums de discussions et des listes de diffusion par exemple, mais là encore il faudra parler C++. Autre élément perturbant est le fait qu'aucune convention de nommage officielle ne soit établie pour la programmation PyQt. Les recommandations Python présentant des incompatibilités avec les conventions de nommage utilisées par Qt, la majorité des projets PyQt sont donc écrits en donnant priorité à ces dernières afin de préserver une certaine homogénéité du code.

Autre critique négative, le retard au niveau de certaines fonctionnalités des environnements de développement intégrés pour Python. Pour avoir utilisé Eric 4 durant la quasi totalité du développement, je peux dire que ses fonctionnalités de complétion syntaxique sont vraiment très rudimentaires. Il a cependant le mérite de présenter des outils tels que de la génération de documentation ou de diagrammes, mais qui sont malheureusement eux aussi

trop rudimentaires pour être utiles. On préfèrera Epydoc pour la génération d'une documentation sérieuse, et un bon outil de modélisation UML tel qu'Umbrello pour les diagrammes. Eric intègre par contre quasi à la perfection les outils de développement Qt.

Python est un langage assez intuitif, ce n'est pas pour rien qu'il est souvent utilisé dans un but pédagogique. Il présente beaucoup d'avantages à l'utilisation comme une lisibilité accrue, un débogage facile grâce aux tracebacks, une auto-documentation, son ramasse-miettes, son typage dynamique, ... Bien que possédant ses propres patterns et sa propre manière « pythonneuse »d'aborder certains cas, Python constitue pour du développement rapide un langage de premier choix.

Qt est un framework qui a déjà fait ses preuves dans bien des domaines, et à l'utilisation on comprend pourquoi. Une fois pris en main et certains concepts assimilé, il permet d'évoluer assez rapidement et d'être très efficace dans un développement. De plus, il permet l'application de concepts importants tels que MVC, afin de produire du code facilement maintenable et dissocié de l'interface utilisateur en elle-même. Qt est également fourni avec un série d'outils complémentaires et de qualité facilitant le travail du programmeur.

8.6 La virtualisation

La virtualisation en entreprise est une notion encore relativement récente, cependant elle est déjà sur le point de bouleverser le monde de l'informatique tel que nous le connaissons ainsi que notre manière de travailler. La virtualisation est donc le sujet du moment dans le monde de l'informatique, il est donc dommage que celui-ci ne soit pas abordé dans un cours à l'IPL.

8.7 XenMonitor et son futur

L'utilisation de XenMonitor permettra, à terme, de répondre à un besoin précis d'Unipress : rendre la technologie utilisable par un public beaucoup plus large. Une fois la gestion de la virtualisation facilitée, son introduction en entreprise en sera grandement facilitée, car il ne faut pas oublier que l'informatique est au service du business, et non pas le contraire.

Vu son statut de logiciel libre et ouvert, XenMonitor sera probablement appelé à évoluer rapidement, permettant ainsi de répondre à des besoins plus larges. Il sera également possible pour moi-même de fournir du support relatif à celui-ci.

9 Bibliographie

- SUMMERFIELD Mark, Rapid GUI Programming with Python and QT : The Definitive Guide to PyQt Programming.
  États-Unis d'Amérique : Pearson Education, 2008, 625p.
- BLANCHETTE Jasmin, SUMMERFILED Mark, Qt 4 et C++ : Programmation d'interfaces GUI.
  Paris : CampusPresse, 2007, 550p.
- MELLOR Ewan, SHARP Richard, SCOTT David, Xen Management API.
  XenSource, 2006-2007, 167p.
- PILGRIM Mark, Plongez au coeur de Python,
  http ://www.diveintopython.org, 2006, 334p.
  Un excellent livre sur Python, publié au format électronique sous licence libre, mais également édité sous forme de livre.
- GUERRA Stanislas, gnu :python :style_guide_fr,
  http ://stan.openmod.org/doku.php/gnu :python :style_guide_fr, dernière mise-à-jour le 16 mars 2007
  Un très bon résumé francophone des normes recommandées par les développeurs Python.
- VAN ROSSUM Guido, WARSAW Barry, PEP 8 – Style Guide for Python Code, http ://www.python.org/dev/peps/pep-0008, 5 juillet 2001
  Le document original reprenant les normes recommandées par les développeurs Python.
- Wikipédia, Virtualisation, http ://fr.wikipedia.org/wiki/Virtualisation
- Wikipédia, Xen, http ://fr.wikipedia.org/wiki/Xen
- Wikipédia, Logiciel libre, http ://fr.wikipedia.org/wiki/Logiciel_libre
- Wikipedia, Comparison of virtual machines,
  http ://en.wikipedia.org/wiki/Comparison_of_virtual_machines
- Wikipedia, Hypervisor, http ://en.wikipedia.org/wiki/Hypervisor
- Wikipedia, Virtualization, http ://en.wikipedia.org/wiki/Virtualization
- Wikipedia, x86 virtualization,
  http ://en.wikipedia.org/wiki/X86_virtualization
- LinuxFR, La virtualisation et le libre : où en est-on ?,
  http ://linuxfr.org/2008/04/28/23997.html, 28 avril 2008
- Trolltech Online Reference Documentation, http ://doc.trolltech.com, Trolltech
  La documentation en ligne de référence fournie par Trolltech.
- The PyQt and PyKDE community Wiki,
  http ://www.diotavelli.net/PyQtWiki
  Un excellent wiki communautaire sur PyQt.

- Qt fr La communauté francophone, http ://www.qtfr.org
  Un site communautaire francophone sur Qt, avec un forum très actif et des tutoriaux.
- Xen wiki, http ://wiki.xensource.com
  Le wiki officiel de Xen.
- libvirt virtualization API, http ://www.libvirt.org
  Le site de la libvirt, une interface commune à Xen, QEMU, KVM, LXC et OpenVZ.
- PyQt, http ://www.riverbankcomputing.co.uk/software/pyqt
  PyQt, l'interface Python de Qt.
- Qwt, http ://qwt.sourceforge.net
  Le site de Qwt, la librairie de dessin de graphique pour Qt.
- PyQwt, http ://pyqwt.sourceforge.net
  Le site de l'interface Python de Qwt.

10 Annexes

Copies d'écran/

Doc/ La documentation de XenMonitor générée par Epydoc.

Installation Windows/ Les fichiers nécessaires afin de déployer XenMonitor sous Windows.

Manuel/ Un document décrivant la procédure de compilation des fichiers du logiciel, son déploiement sous Debian GNU/Linux et Windows XP, la configuration de Xen à effectuer ainsi que son utilisation. Au format PDF, HTML et les sources Latex.

Mémoire/ Le présent document. Au format PDF, HTML et les sources Latex.

Suppléments/ Des fichiers relatifs à certains sujets traités dans ce document, tels que l'image VMWare project-open, WindowsXenPV ou encore de la documentation libre de distribution.

XenMonitor/ Les fichiers sources et compilés du logiciel, prêts à être exécutés.

Jeux de tests.odt Les résultats d'une série de jeux de tests concernant XenMonitor.

Scénarios.odt Un scénario d'utilisation de XenMonitor permettant de compléter l'analyse.