ETNA
Projet de fin d'étude 2005-2007
RIMELSE
Production Plan

Table des matières

Chapitre 1

Introduction

Nowadays, computing solutions users are more nomad. But, they are also confront to a new problem : how to have an operating system which agree to their needs without a laptop ? Now, free software's solutions provide mobile solutions like "live cd" or "USBlive" keys systems. Yet, this solutions still have incinvenients like data access rapidity on the source media (ex : the live CD disk). But also the created data persistence when using this solutions and at least the impossibilty to personalize you own system.

Dans le cadre de leur projet de fin d'étude ETNA (PFE), the Else group, ETNA Linux System Engineers consist of six studients who have decided to provide a solution for the problem quoted previously, so the RIMELSE distribution is the subject of the present production plan.

This production plain include, in first, a description of the distribution's differents caracteristics like the Run In Memory and the "evoluability" gestion. In a second time,it discribe the diferents realisations of the ELSE group in order to build the RIMELSE distribution. And finaly, the technical choices made by the development team.

Chapitre 2

Presentation of the project

2.1 An "Evoluable" Linux Distribution

To give solutions for the quoted problems we have created "RimElse GNU/Linux".

"RimElse GNU/Linux" is a "GNU/Linux Distribution" "Run In Memory". First, let's see what is a GNU/Linux distribution.

The wikipedia.com[1] definition is very talking :

> A typical Linux distribution comprises a Linux kernel, GNU tools and libraries, additional software, documentation, a window system, window manager, and a desktop environment. Most of the included software is free software/open-source software which is distributed by its maintainers both as pre-compiled binaries and in source code form, allowing users to modify and compile the original source code if they wish. Other software included with some distributions may be proprietary and may not be available in source code form.
>
> Many provide an installation system akin to that provided with other modern operating systems. Self-hosting distributions like Gentoo Linux and Linux From Scratch provide the source code of all software and include binaries only of a basic kernel, compilation tools, and an installer ; the installer compiles all the software for the specific microarchitecture of the user's machine.

[1] http ://en.wikipedia.org/wiki/Linux_distributions

> Distributions are normally segmented into packages, each package holding a specific application or service ; examples of packages include a library for handling the PNG image format, a collection of fonts, or a web browser. The package is typically providing as compiled code with installation and removal of packages handled better than a simple file archiver. This software is said to be the package management system (PMS) of the distribution. Each package intended for such a PMS contains meta-information like description, version, "dependencies", etc. The package management system can evaluate this meta-information to allow package searches, to perform an automatic upgrade to a newer version, to check that all dependencies of a package are fulfilled and/or to fulfill them automatically. Package management systems include :

The run in memory is based on a simple idea, the complete system is loaded in memory and executed from there.

A lot of like that already exist, but we offer to bring a non negligible favour, create an “Evoluable” system, this word is the alliance between Evolution and Malleable.

Evolution, because we offer to user the possibility of change his softwares versions and, add or delete ones like a classical distribution.

Malleable because we want that system be abble to answer to each users needes, but also to satisfy the most poepole. Without omit the fact that the system is “Run In Memory” and so optimized.

“RimElse GNU/Linux” must be your pocket system !

2.2 Employment

It’s easy to present concepts without having to determine the “employment”.

RIMELSE will be usable in a lot of context, we are about of decribing two of theim.

The first is a ETNA’s studient.

2.2.1 *etna_ e* is a Etna's studient

Be an ETNA's studient gives some constraints for his system :
- Have a link in order to be connect at the intranet,
- Have a flash player for his browser,
- Have a SSH client to be abble to connect school computers,
- Have a C compiler and debugger to work on projects,
- etc ...

A studient working on an another computer don't have all his tools. If he doesn't have a laptop, he couldn't work if he doesn't use his computer.

The interest of the RIMELSE distribution is that with a USB KEY (and a CDROM if the guest computer's motherboard doesn't support USB boot), the studient *etna_ e* will have his own work environment with him and keep his work and differents documents if the USB key storage is enought.

FIG. 2.1 – Schéma d'utilisation pour un étudiant

2.2.2 *adm_p* is a computer science teacher

adm_p need to make a C or Unix lesson to his studients, but the school doesn't have UNIX computers.

With a RIMELSE, he have adapt to his needed with samba client, C complier, he can load a unix systyem on all computers of the class with only one USB key prepared before.

FIG. 2.2 – Schéma d'utilisation pour un professeur

Chapitre 3

The RIMELSE goals

The ELSE team have determined some objectives for the RIMELSE project. The first of them is, to have a fast ant the most effective in peripherals cheking and starting sequence.

Then RIMELSE will be abble to run only in RAM, without having to acces to the media from which the system have been loaded.

A system synchronization must be possible when the user wants, in order to save his modifications.

RIMELSE, like any other distribution have his own configuration tools and binaries packages management. That's why our goal is to improve the dependences management by providing the possibilty to done binaries patchs. So for adding a support for a package, for example a LDAP support for Postfix, it only be needed to apply a patch and don't to delete the package and install the one which includes the support. It's why whe must redefine a binarie package type which be temporarily called .reb (RimElse Binaries). Those binaries must been optimized like for a embarked system, in the goal that the distribution become the lightest possible.

And finaly the last point we will provide a personalized system generation tool. Everyone will be able to generate his own system according with his needed. The $etna\_e$ doesn't have the same needed of the adm_p teacher but both need the RIMELSE.

3.1 The realistations for RiMelse

3.1.1 The realisation

To reach our goals, many steps have been etablished.

The starting sequence

Fisrt of all it's necessary to optimize the starting sequence, it will result from this the creation of a peripherals recognition script the lightest and most efficent possible. It's mean that the peripherals majority must be recognized.

Always by the fact that the distribution must be fast, we must redefine the services starting utility.

The binaries gestion

RIMELSE is a Run In memory distribution, that imply a special binaries treatment. So, it have been decided to define a new packages format and an adapted manager.

The configuration's manager

The configuration manager will allow to the user to modify his distribution parameters fallowing his needed, for example the grafical interface or network configuration. So it will be linked with the packages manager.

The distribution's generator

To optimize the distribution personalisation, it would be possible for the user to generate his own RimElse according to his needed.

RIMELSE RimElse proposera plusieurs type de profils prédéfinis, par exemple un profil étudiant ETNA ou encore développeur Java.

Enfin, pour être en adéquation avec les différents points cités et pour répondre aux problématiques posés, la RimElse sera disponible dans différents supports définis tels que les compact flash, CD-Rom et au seins de partages réseaux. Ce chapitre explique ce qu'on veut apporter avec RimElse.

3.2 The technical choices

Prior to the request concerning legibility, disponibility ans especially on continuniation, the following choices were found essential.

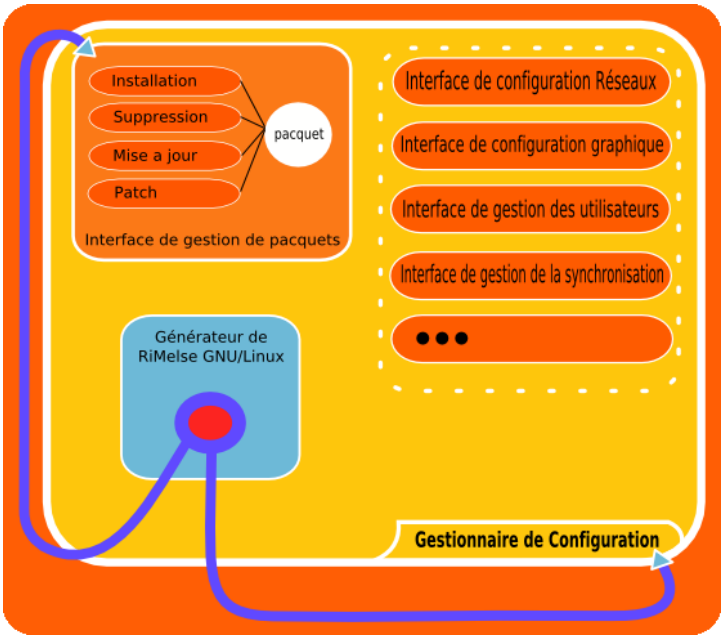

FIG. 3.1 – Schéma montrant l'agancement générale de la configuration

Concerning documentation, Two tools held our attention.
- On the one hand LaTeX, for the project's documentation part.
- On the other hand Doxygen which will generate documentation concerning the source code.

These tools allow to return documents under various formats like standard API's documentation.

In addition, any modification of the source code, or on the documentation must be able to be followed by the whole group. We have decided to deploy "git", a powerfull Source Code Manager (SCM). In particular then used for the management of the source code for the Kernel Linux, it allows to warn informed the group of any modification.

Beyond the control aspect, this work process allows a total implication of each member, whathever his involvement. Indeed it can be in the project process to retrogress if an error were added in the code.

Regarding to the various developments we meaningly chose two languages :

- Ruby which is a combination of SmallTalk for the utilisation, Python for the simplicity and Perl for the flexibility.
- C, impossible to circumvent because of tis presence in the Linux Kernel.
- We should have to set shell scripts.

Those differents choices leadto have a "perene" solution, but also to answer the poblems of lisibility and availabilities.

Concerning the Linux core, we'll only work on version 2.6 or higher, because the newest cores are the most provided in hardware supports.

Chapitre 4

Description techinique de RIMELSE

Chapitre 5

Conclusion

Bibliographie

[1] Définition de distribution : http ://fr.wikipedia.org/wiki/Distribution_Linux

[2] LaTeX : http ://www.latex-project.org/

[3] Doxygen : http ://www.stack.nl/~ dimitri/doxygen/

[4] Git : http ://www.kernel.org/git/

[5] Ruby : http ://www.ruby-lang.org/en/