ABout Time Linux - the design (TRUNK)

Eric D. Schabell
erics@abtlinux.org

June 15, 2006

Abstract

This document will detail the design of the *ABout Time Linux* package manager. It expands on the earlier work done in the requirements phase and will focus on creating the elements needed to manage software on a *ABout Time Linux* system.

1 Introduction

The *ABout Time Linux (AbTLinux)* package managers design will be dealt with in this document. I will cover the details of the *AbTLinux* design and provide a general overview in the form of *UML* diagrams.

The driving force behind this project is to design a basic software package management system that will provide the needed infrastructure to not only install software packages, but also be able to maintain the systems it is installing the software packages onto. I believe this can be done using source based packages thereby avoiding the existing RPM and DEB package management systems. This is not a new idea, nor do I pretend to have found the Holy Grail of package management tools. The primary goal is a well documented design, with clearly documented coding practices which will result in an easily maintainable package manager. This in turn will be the foundation of this Linux distribution. Make no mistake about it, the clarity of design and coding practices will take the foremost priority in this project.

I will present my ideas in the following sections which will include implementation choices such as the language and system requirements. This design is based on earlier work in (Schabell, 2006).

2 Design

Here the design and implementation choices made for *AbTLinux* will be laid out.

2.1 Package structure

A single package will have one file containing the entire structure needed for installing the software it offers. This structure will be split into sections, such as the following:

- details{}
- pre{}
- configure{}
- pre-build{}
- build{}
- pre-install{}
- install{}
- post{}

2.1.1 Example - fortune

To provide a simple overview of a package description we will detail a package. The running example will be of *fortune*, a software application that prints a random, hopefully interesting, adage.

2.1.2 Install locations

Base system for AbTLinux will be in /usr, all others will be installed into /usr/local/*. This will facilitate ease of usage within other Linux systems that follow the LSB/FHS. Initial implementaion will allow for setting the install location, therefore we can install everything into /usr/local/* for ease of testing on all available Linux distributions.

2.2 Configuration update tools

Here we mean tools dealing with how we want configuration file updates to be handled. Almost everything located in /etc of the Linux filesystem is considered holy to a running system. They need to be updated sometimes when a new package version is installed, but should never destroy an existing configuration. We want to provide for the following:

- view existing configuration file
- view new (default to install) configuration file
- allow editing to take place in old or new config file

- show user diff
- let user select one of the above options to install

2.3 Package manager

The AbT API outline can be found online at the AbTLinux website. The basic outline can be seen in Figure 1.

2.3.1 Repository names

Providing the following repositories as subversion tags:

1. **HEAD** - the current working repository.
2. **TAGGED-RELEASE** - released version tagged with a unique name.

2.3.2 File storage

The various files that are generated and needed by the abt package manager will be stored in the following locations:

- package sources - `/var/spool/abt/`
- install info - `/var/state/abt/install/PKG_NAME/`
- cached info - `/var/state/abt/cached/PKG_NAME/`
- frozen info - `/var/log/abt/frozen.log`
- installed log - `/var/log/abt/installed.log`
- journal - `/var/log/abt/abt.log`
- newfeed - `/var/log/abt/news.log`
- install queue - `/var/log/abt/install.queue`
- tracking info - `/etc/abt/tracking.cfg`

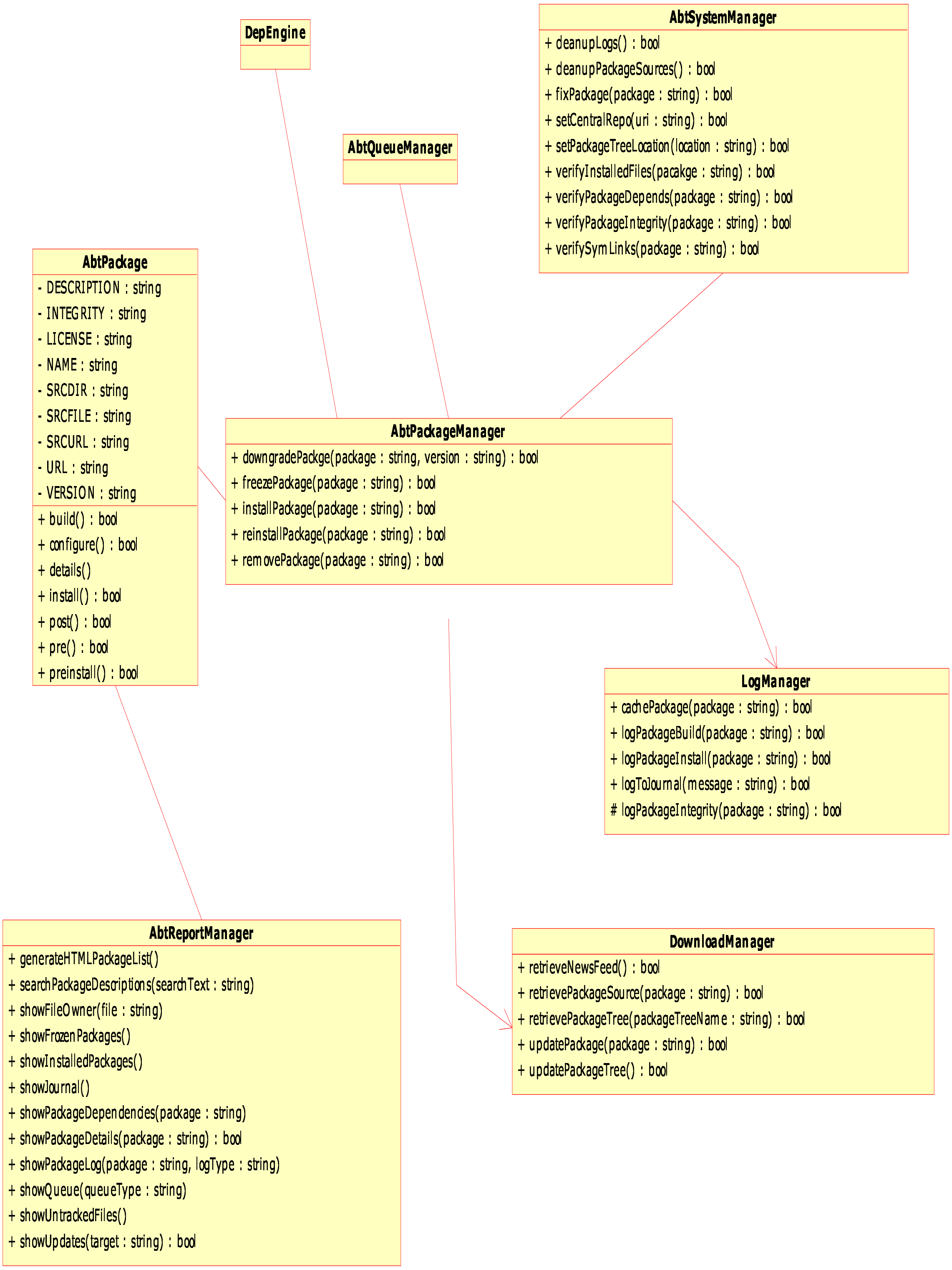
DepEngine
AbtQueueManager
AbtSystemManager
+ cleanupLogs() : bool
+ cleanupPackageSources() : bool
+ fixPackage(package : string) : bool
+ setCentralRepo(uri : string) : bool
+ setPackageTreeLocation(location : string) : bool
+ verifyInstalledFiles(pacakge : string) : bool
+ verifyPackageDepends(package : string) : bool
+ verifyPackageIntegrity(package : string) : bool
+ verifySymLinks(package : string) : bool
AbtPackage
- DESCRIPTION : string
- INTEGRITY : string
- LICENSE : string
- NAME : string
- SRCDIR : string
- SRCFILE : string
- SRCURL : string
- URL : string
- VERSION : string
+ build() : bool
+ configure() : bool
+ details()
+ install() : bool
+ post() : bool
+ pre() : bool
+ preinstall() : bool
AbtPackageManager
+ downgradePackge(package : string, version : string) : bool
+ freezePackage(package : string) : bool
+ installPackage(package : string) : bool
+ reinstallPackage(package : string) : bool
+ removePackage(package : string) : bool
LogManager
+ cachePackage(package : string) : bool
+ logPackageBuild(package : string) : bool
+ logPackageInstall(package : string) : bool
+ logToJournal(message : string) : bool
# logPackageIntegrity(package : string) : bool
AbtReportManager
+ generateHTMLPackageList()
+ searchPackageDescriptions(searchText : string)
+ showFileOwner(file : string)
+ showFrozenPackages()
+ showInstalledPackages()
+ showJournal()
+ showPackageDependencies(package : string)
+ showPackageDetails(package : string) : bool
+ showPackageLog(package : string, logType : string)
+ showQueue(queueType : string)
+ showUntrackedFiles()
+ showUpdates(target : string) : bool
DownloadManager
+ retrieveNewsFeed() : bool
+ retrievePackageSource(package : string) : bool
+ retrievePackageTree(packageTreeName : string) : bool
+ updatePackage(package : string) : bool
+ updatePackageTree() : bool

Figure 1: The abt package manager UML class diagram

3 Dictionary

A list can be found here that contains terms and their AbTLinux related definitions. Hopefully this will keep everyone on the same page.

- **Package** - this will contain all needed information for the abt tool to install a single piece of software.
- **abt** - the package manager tool that can be run from the command line and provides for all software management on the AbTLinux distribution.
- **depEngine** - the dependecny engine, part of the tool set used by 'abt' to facilicate package management.
- **Dependency** - a package X will have a depenceny on another package Y if X needs a service provided by Y. X is then dependent on Y.
- **AbTLinux** - the name of this distribution, derived from ABout Time Linux.
- **Journal** - the running log of abt actions, where 'abt' will provide information as to actions taken on the AbTLinux system.

References

Schabell, E. (2006). ABout Time Linux – the requirements. Technical report, Radboud University Nijmegen, Nijmegen, The Netherlands, EU.