EDOS-Tools Tutorial: EDOS Tools for Linux Distributions Dependencies Management and Quality Assurance

François Déchelle[1], Fabio Mancinelli[2]
[1]EDGE-IT, France
fdechelle@mandriva.fr
[2]PPS - Université Paris VII, France
fabio@pps.jussieu.fr

Abstract. Free and Open Source Software (FOSS) distributions are the results of the effort of third party actors in collecting independently developed software products, in a consistent and usable form. The widespread adoption of these distributions as infrastructural components in many strategic contexts of the information technology society has drawn the attention on the issues regarding how to handle the complexity of assembling and managing a huge number of (packaged) components and how to guarantee their quality. This tutorial will describe how the EDOS project has tackled these issues. First it will describe the problems related to the quality assurance of Linux distributions and will present the tools that have been developed to manage testing process. It will then introduce the problems that occur when managing inter-package relations in large package repositories and will showcase tools that can be used to analyze and manage large package repositories.

Description

The tutorial will be organized in two parts. The first part will introduce the issues related to testing and quality assurance of heterogeneous Open Source packages and present tools developed in the EDOS project for managing Linux distribution testing and quality assurance processes. The second part will introduce the state of the art in Linux package management systems and problems regarding the management of inter-package relations (dependencies and conflicts) in large package repositories.

1.1 Testing and quality assurance

The tutorial will detail the use of the following tools:

- **Testrunner**: a tool for conducting automatic and manual tests and reporting test results. Testrunner uses an XML-based test specification and can report test results using several reporting plug-ins, for instance to report results to the QA portal using HTTP request.
- **TULIP**: a tool to test upgrades of Linux installations using virtual machines and the distribution standard upgrade tools. TULIP can run automatic upgrades of installed Linux distributions, test the upgraded distributions and reports results to the QA portal using Testrunner.
- **QA Portal**: a web portal for test management, that allows testers and distribution developers to have a real-time and accurate view of the distribution testing process including available test suites, tests, reports of executed tests...

The tutorial will present how to install the tools, how to setup a complete distribution testing environment and will feature a hands-on session on a realworld distribution testing process.

1.2 Large package repositories complexity and dependency management

The second part will introduce the state of the art in Linux package management systems and problems regarding the management of inter-package relations (dependencies and conflicts) in large package repositories. A set of tool that can be used by distribution editors to analyze and manipulate repositories in order to find potential problems due to incorrect inter-package relation specifications will then be showcased:

- **DEB/RPMCheck**: a dependencies correctness checker. DEB/RPMCheck provides a fast way for analyzing whole package repositories and to spot problems that can be present in package dependency meta-data.
- **History**: historical analysis and symbolic manipulation of package repositories. History is powered by a powerful functional language called DQL that enables the user to perform sophisticated queries on package repositories and to manipulate them in a declarative way by using some advanced operators. Moreover, History supports the analysis of historical data for tracking the evolution of package repositories over time.
- **Anla**: a web service for package repository exploration. Anla is the web-oriented counterpart of History that can be used by distribution editors to provide a direct feedback on the distribution status to users, testers and developers. Advanced queries can be performed using this interface and hyperlinked graphical results are provided as output.
- **Tart**: an optimized “thinner” for building custom distributions. Tart enables distribution editors to build custom distributions that met some constraints (e.g. space or priority). By using Tart it is possible to create package sets that are closed with respect to dependency relations and that satisfy the optimization needs defined by the constraints.