The Ultimate Debian Database
Consolidating Bazaar Metadata for Quality Assurance and Data Mining

Lucas Nussbaum* and Stefano Zacchiroli†

* LORIA, University of Nancy, France
† Université Paris Diderot, PPS, France
Most FLOSS projects require some infrastructure

- Often similar needs
- Led to the development of standard *forge*
FLOSS needs: distribution vs development

Most FLOSS projects require some infrastructure
- Often similar needs
- Led to the development of standard forges

FLOSS distributions: more complex requirements
- Archive management, build bots, Quality Assurance tools
- No standard solution:
  - Small number of different distributions
  - Different history, backgrounds, processes

⇒ No interest in converging to a standard solution
FLOSS needs: distribution vs development

Most FLOSS projects require some infrastructure
- Often similar needs
- Led to the development of standard forges

FLOSS distributions: more complex requirements
- Archive management, build bots, Quality Assurance tools
- No standard solution:
  - Small number of different distributions
  - Different history, backgrounds, processes
⇒ No interest in converging to a standard solution

However: Need to combine data inside and between distributions
- Research: focus on data-mining rather than on implementations
- Collaboration between distributions
- Quality Assurance
Data-mining for Quality Assurance

Debian: most important community-driven distribution
27’000 software packages

- Need to identify packages of sub-standard quality

Which critical bugs affect source packages in the testing branch, sorted by decreasing package popularity?
Data-mining for Quality Assurance

Debian: most important community-driven distribution
27’000 software packages

- Need to identify packages of sub-standard quality

Which critical bugs affect source packages in the testing branch, sorted by decreasing package popularity?

- Need to combine data about packages, bugs, packages popularity
- Requires a 200+ lines script ← really hard
Data-mining for Quality Assurance

Debian: most important community-driven distribution
27'000 software packages

- Need to identify packages of sub-standard quality

Which critical bugs affect source packages in the testing branch, sorted by decreasing package popularity?

- Need to combine data about packages, bugs, packages popularity
- Requires a 200+ lines script \(\leftarrow\) really hard

this presentation:
Ultimate Debian Database
The countermeasure developed inside the Debian project to enable data mining and Quality assurance
Agenda

1. The Debian Data Hell
   - Why is it so hard to combine data in Debian?
   - Why are we in that situation?

2. Ultimate Debian Database: Architecture and Current Status

3. Examples

4. Conclusions
Several factors contribute to this situation:

- Heterogeneity
- Community inertia
- Services organization
# Heterogeneity

Debian infrastructure composed of several different services

- Developed over 16 years, by different people
- With different design choices and technologies

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Name</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Archive mgmt</td>
<td>dak</td>
<td>Python, PostgreSQL. Text export</td>
</tr>
<tr>
<td>build daemons</td>
<td>wanna-build</td>
<td>Perl, PGSQL. was BDB</td>
</tr>
<tr>
<td>Bug tracker</td>
<td>debbugs</td>
<td>Perl, mbox files</td>
</tr>
<tr>
<td>devel accounts</td>
<td>Debian LDAP</td>
<td>LDAP, GnuPG keyrings</td>
</tr>
<tr>
<td>Devel tracking</td>
<td>carnivore</td>
<td>Perl, Berkeley DB.</td>
</tr>
<tr>
<td>Packages uploads</td>
<td></td>
<td>Mailing list archives.</td>
</tr>
<tr>
<td>Upstream monitoring</td>
<td>DEHS</td>
<td>PHP, PostgreSQL database</td>
</tr>
<tr>
<td>Package popularity</td>
<td>popcon</td>
<td>Perl, text dump</td>
</tr>
<tr>
<td>Policy conformance</td>
<td>lintian</td>
<td>Perl, text dump</td>
</tr>
<tr>
<td>Packages tagging</td>
<td>debtags</td>
<td>C++, text DB</td>
</tr>
<tr>
<td>Package dashboard</td>
<td>packages.qa</td>
<td>Python, XSLT</td>
</tr>
</tbody>
</table>
Community Inertia

Why not **rewrite some of those services?**

- There is no central authority to take such decisions

- Developers often like their original design
  Cared about solving their own problem, not about the "greater good"

- The developer community doesn’t feel there is a problem:
  Only a few teams experience the data hell
  = the teams that need to combine data
Services organization

Debian infrastructure and hosting is provided through sponsoring

⇒ Services are scattered over the planet

But:
- Services are tightly coupled
- Make their data publicly available

⇒ A lot of hidden inter-service dependencies

A lot of resistance to change:
changing a service always breaks other services
Services organization

Debian infrastructure and hosting is provided through sponsoring

⇒ Services are scattered over the planet

But:

● Services are tightly coupled
● Make their data publicly available

⇒ A lot of hidden inter-service dependencies

A lot of resistance to change:
changing a service always breaks other services

Ultimate Debian Database

⇒ Tries to Provide a solution to that, based on a single PostgreSQL DB

⇒ But needs to play by the rules of a community-driven distribution
   In particular : does not try to replace services, only replicate data
UDD: Architecture

Debian derivatives (Ubuntu, Skolelinux, ...)

Package Archive

Package Popularity

Bug Tracking System

Package Archive

Package Popularity

Developer DB (LDAP)

Ultimate Debian Database

Gatherer

Gatherer

Gatherer

Gatherer

Gatherer

Gatherer

Ui

Query

Human users, Data mining, ...
UDD : Design choices

- General-purpose tool, user-friendly
  - No specific applications in mind ➔ Less optimization opportunities 😞
  - No surrogate keys
UDD : Design choices

- General-purpose tool, user-friendly
  - No specific applications in mind
    → Less optimization opportunities 😞
  - No surrogate keys

- Inconsistencies are preserved
  - They might be interesting in some cases
  - When too problematic: hidden using SQL VIEW

Lucas Nussbaum and Stefano Zacchirolı  The Ultimate Debian Database
**UDD : Design choices**

- General-purpose tool, user-friendly
  - No specific applications in mind → Less optimization opportunities 😞
  - No surrogate keys

- Inconsistencies are preserved
  - They might be interesting in some cases
  - When too problematic: hidden using SQL VIEW

- Full data reloads instead of incremental updates
  ⇒ Improves correctness
UDD: Design choices

- General-purpose tool, user-friendly
  - No specific applications in mind
    - Less optimization opportunities 😞
  - No surrogate keys

- Inconsistencies are preserved
  - They might be interesting in some cases
  - When too problematic: hidden using SQL VIEW

- Full data reloads instead of incremental updates
  ⇒ Improves correctness

- (No) storage of historical data
  - Except for some aggregate values
    number of packages, number for each VCS/format
  - But full DB dump is available
Current Status

17 gatherers:
- Source and binary packages for Debian and Ubuntu
- Debian bugs
- Ubuntu bugs
- Packages popularity for Debian and Ubuntu
- Full history of uploads
- Upstream status (DEHS) – is the package up-to-date?
- Developers identities (carnivore)
- Official developers (Debian LDAP)

And packages tags (debtags), policy conformance (lintian), build daemons status (wanna-build), orphaned packages, full history of packages removals, screenshots, localized packages descriptions, new packages under review, status regarding testing migrations, ...

60 tables, 7 millions of tuples, 3.8 GB
Examples

- Popular yet buggy packages:

```sql
select sources.source, id, insts, title
from bugs
join sources on sources.source = bugs.source
join sources_popcon
  on sources_popcon.source = bugs.source
where severity >= 'serious'
  and distribution = 'debian'
  and release = 'squeeze'
  and affects_testing
order by insts desc
```

- Tracking of neglected packages and neglecting developers
- Collaboration with Ubuntu
- MSR’2010 challenge
Conclusions

Ultimate Debian Database:
- Attempt at solving the *Debian Data Hell*
- Originally developed for QA and collaboration with derivatives
- Should also ease academic research on Debian
- Makes Debian a better citizen of the FLOSS ecosystem by providing access to most of its data

http://udd.debian.org/