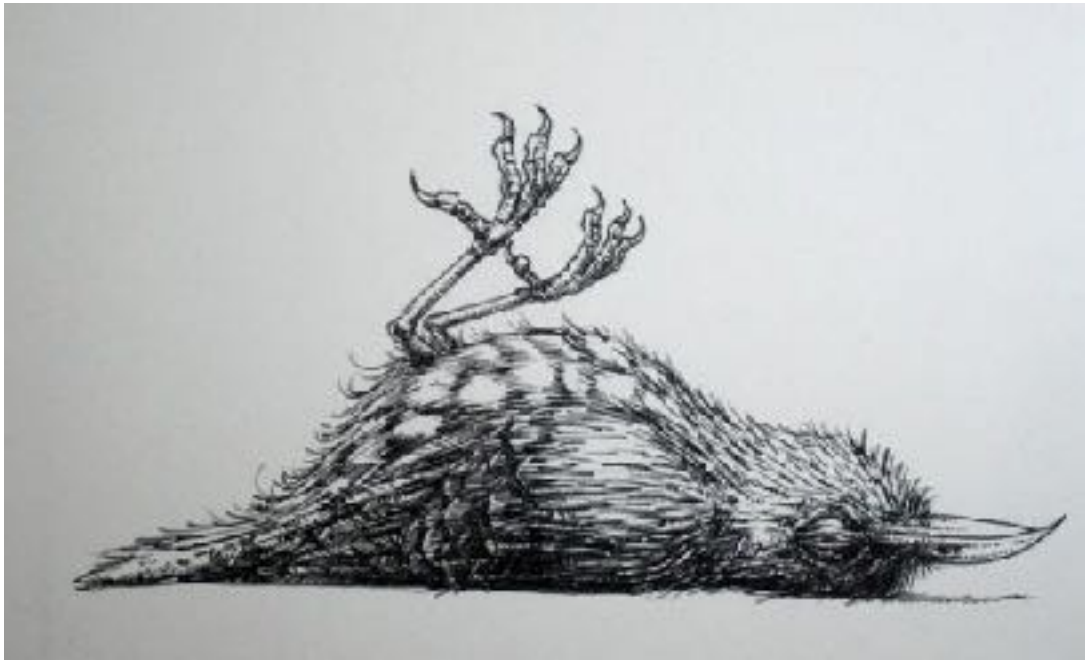
Modelos de Gobernanza en Proyectos de Software Libre

Jordi Cabot & Javier Cánovas

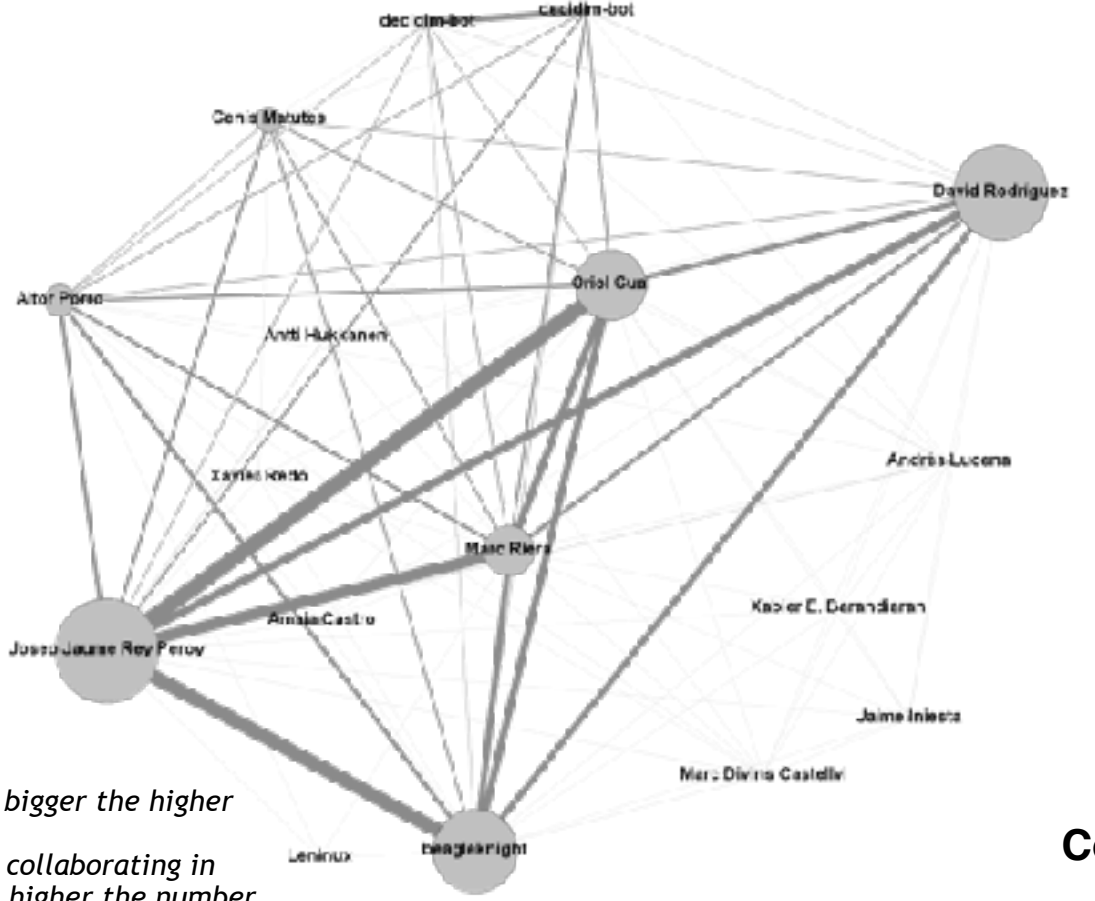
¿Cómo puedo asegurar el éxito de mi proyecto?



Obj: Discutir estrategias para asegurar la sostenibilidad del SW Libre

Análisis de Software

Decidim platform



*A **node** represents a user, the bigger the higher number of edited files is*
*An **edge** represents two users collaborating in the same file, the thicker the higher the number of files*

Collaboration Graph (plain)

Bus Factor


“Number of key developers who would need to be incapacitated (hit by a bus), to send the project into disarray that it would not be able to proceed”

The repo_decidim project

The Bus Factor is:

3

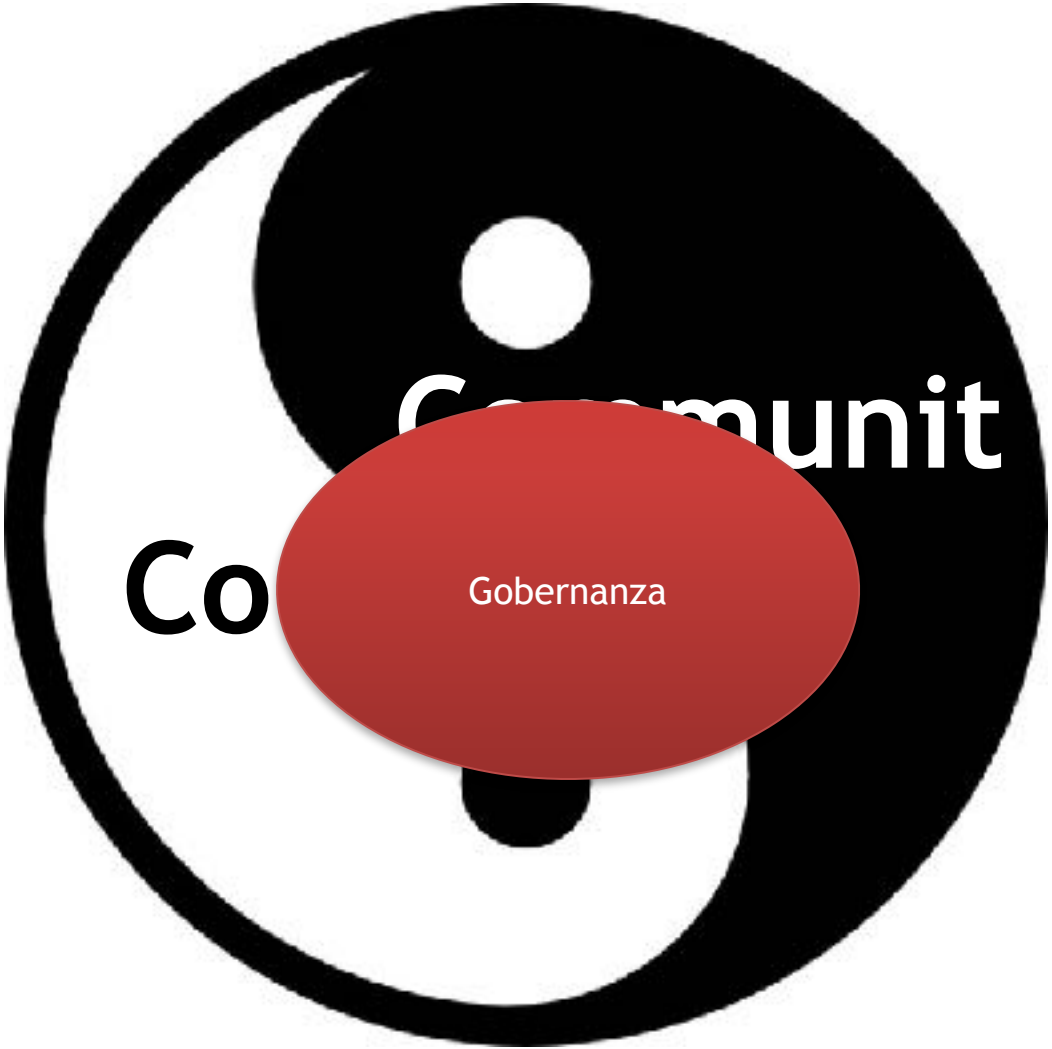
Start Bus Hilling

Summary

In your project 17 developers contributed actively in 1812 files (the most important file extensions are **rb (57.28%), erb (16%) and scss (7.34%)**). The project will have a hard time if **David Rodriguez** is hit by a bus. The project also relies on **beagleknight** and **Josep Jaume Rey Peroy**. In any case, the project can manage without **Oriol Gual**, **Marc Riera**, **Aitor Porro**, **decidim-bot**, **Genis Matutes**, **Antti Hukkanen**, **Marc Divins Castellvi**, **Xavier Redó**, **Andrés Lucena**, **Amaia Castro**, **Leninux**, **GitHub**, **Jaime Iniesta** and **Xabier E. Barandiaran**.

User Relevance

David Rodriguez 30.63%	beagleknight 18.98%	Josep Jaume Rey Peroy 17.52%	Oriol Gual 8.04%	Marc Riera 5.66%
Aitor Porro 5.11%	decidim-bot 3.36%	Genís Matutes 2.41%	Antti Hukkanen 0.13%	Marc Divins Castellvi 0.12%
Xavier Redó 0.1%	Andrés Lucena 0.07%	Amaia Castro 0.07%	Leninux 0%	GitHub 0%
Jaime Iniesta 0%	Xabier E. Barandiaran 0%			


Co
munit
Gobernanza

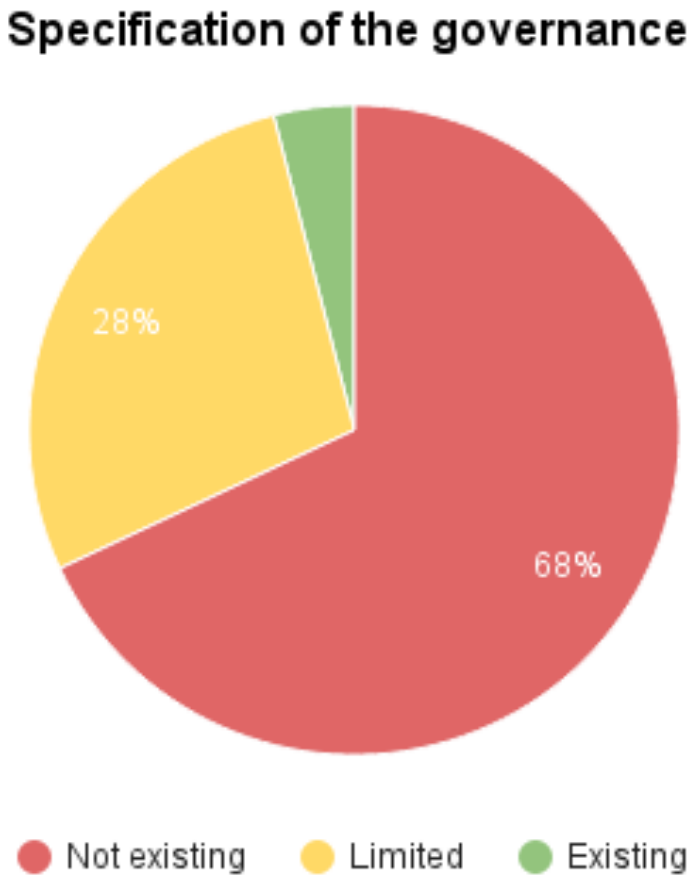
dle.rae.es/?id=JHRSmFV

es Cheat sheets for the / ISI JCR - Journal Citat Notifications Notifica PDF to Word Convert Endalia UOC Web

gobernanza

1. f. Arte o manera de gobernar que se propone como objetivo el logro de un desarrollo económico, social e institucional duradero, promoviendo un sano equilibrio entre el Estado, la sociedad civil y el mercado de la economía.

2. f. desus. Acción y efecto de gobernar o gobernarse.



Los proyectos no son transparentes: No gobernanza explícita

CVE-2014-6412 - WordPress (all versions) lacks CSPRNG

From: Scott Arciszewski <scott () arciszewski me>
Date: Tue, 10 Feb 2015 11:50:16 -0500

Ticket opened: 2014-06-25
Affected Versions: ALL
Problem: No CSPRNG
Patch available, collecting dust because of negligent (and questionably competent) WP maintainers

On June 25, 2014 I opened a ticked on WordPress's issue tracker to expose a cryptographically secure pseudorandom number generator, since none was present (although it looks like others have tried to hack together a band aid solution to mitigate php_mt_seed until WordPress gets their "let's support PHP < 5.3" heads out of their asses).

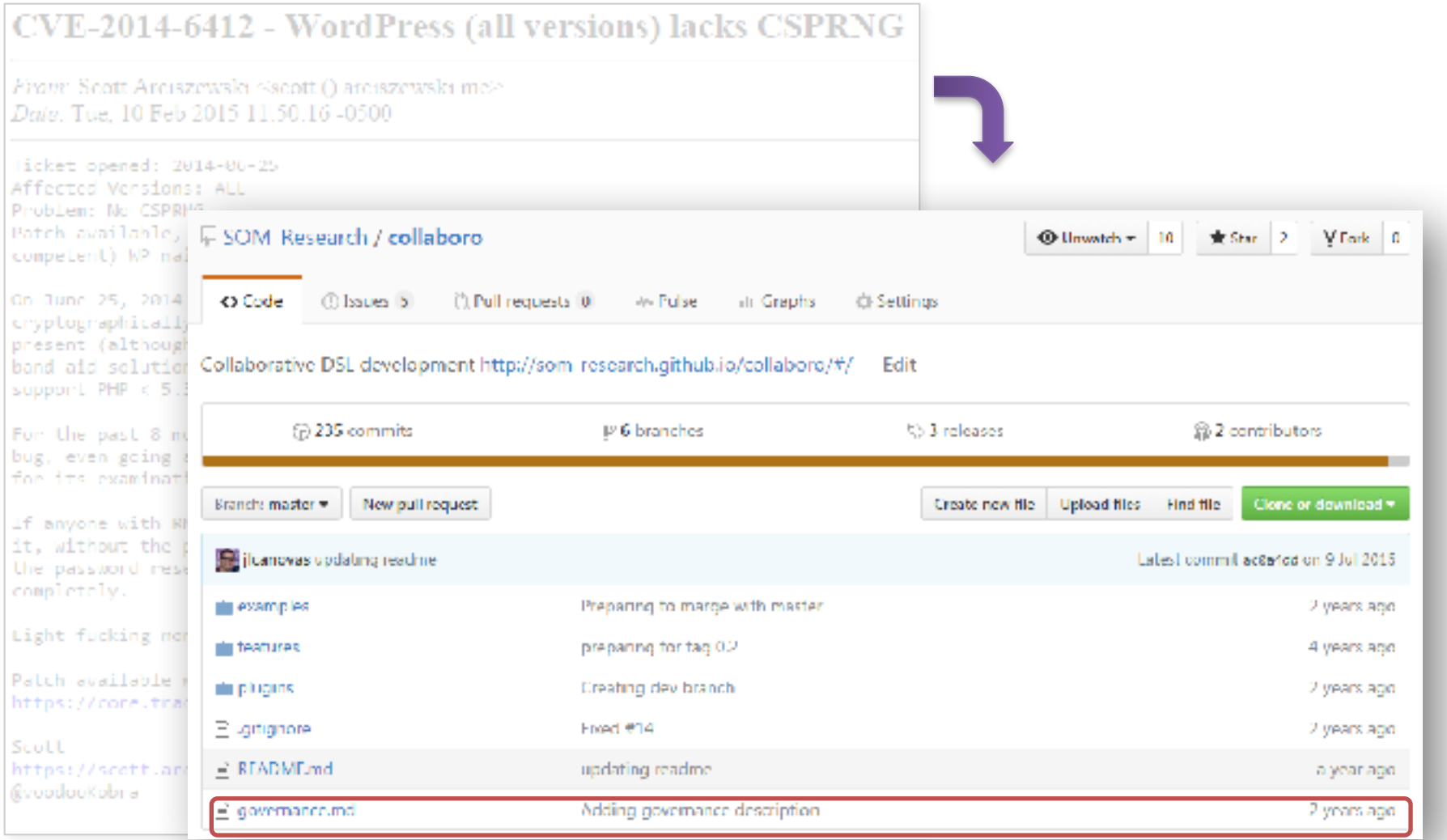
For the past 8 months, I have tried repeatedly to raise awareness of this bug, even going as far as to attend WordCamp Orlando to troll^H^H advocate for its examination in person. And they blow me off every time.

If anyone with RNG breaking experience (cough solar designer cough) can PoC it, without the patch I've provided you should be able to trivially predict the password reset token for admin users and take over any WordPress site completely.

Light f******g months.

Patch available with unit tests and PHP 5.2 on Windows support at
https://core.trac.wordpress.org/attachment/ticket/28643/28643.5.patch

Scott
https://scott.arciszewski.me
@voodooKobra


CVE-2014-6412 - WordPress (all versions) lacks CSPRNG
SOM Research / collaboro
Unwatch 10
Star 2
Fork 0
Code
Issues 5
Pull requests 0
Pulse
Graphs
Settings
Collaborative DSL development http://som-research.github.io/collaboro/#/
Edit
235 commits
6 branches
3 releases
2 contributors
Branch: master
New pull request
Create new file
Upload files
Find file
Clone or download
jlcanovas updating readme
Latest commit ac8e1d on 9 Jul 2015
examples
Preparing to merge with master
2 years ago
features
preparing for tag 0.2
4 years ago
plugins
Creating dev branch
2 years ago
.gitignore
Fixed #14
2 years ago
README.md
updating readme
a year ago
governance.md
Adding governance description
2 years ago

Problema con la Gobernanza

- No evidencia empírica de cuál es el mejor modelo

- Depende de: tamaño, madurez, dominio, filosofía,…

Gobernanza en Fundaciones SW

There is exactly one person who can merge patches into the mainline kernel repository: Linus Torvalds. But, of the over 9,500 patches which went into the 2.6.38 kernel, only 112 (around 1.3%) were directly chosen by Linus himself. The kernel project has long since grown to a size where no single developer could possibly inspect and select every patch unassisted. The way the kernel developers have addressed this growth is through the use of a lieutenant/system built around a chain of trust.

The kernel code base is logically broken down into a set of subsystems: networking, specific architecture support, memory management, video devices, etc. Most subsystems have a designated maintainer, a developer who has overall responsibility for the code within that subsystem. These subsystem maintainers are the gatekeepers (in a loose way) for the portion of the kernel they manage; they are the ones who will (usually) accept a patch for inclusion into the mainline kernel.

Subsystem maintainers each manage their own version of the kernel source tree, usually (but certainly not always) using the git source management tool. Tools like git (and related tools like quilt or mercurial) allow maintainers to track a list of patches, including authorship information and other metadata. At any given time, the maintainer can identify which patches in his or her repository are not found in the mainline.

When the merge window opens, top-level maintainers will ask Linus to "pull" the patches they have selected for merging from their repositories. If Linus agrees, the stream of patches will flow up into his repository, becoming part of the mainline kernel. The amount of attention that Linus pays to specific patches received in a pull operation varies. It is clear that, sometimes, he looks quite closely. But, as a general rule, Linus trusts the subsystem maintainers to not send bad patches upstream.

Subsystem maintainers, in turn, can pull patches from other maintainers. For example, the networking tree is built from patches which accumulated first in trees dedicated to network device drivers, wireless networking, etc. This chain of repositories can be arbitrarily long, though it rarely exceeds two or three links. Since each maintainer in the chain trusts those managing lower-level trees, this process is known as the "chain of trust."

Clearly, in a system like this, getting patches into the kernel depends on finding the right maintainer. Sending patches directly to Linus is not normally the right way to go.

https://www.kernel.org/doc/html/v4.11/process/2.Process.html

Gobernanza en Fundaciones SW

DECISION MAKING

Projects are normally auto governing and driven by the people who volunteer for the job. This is sometimes referred to as "do-ocracy" -- power of those who do. This functions well for most cases.

When coordination is required, decisions are taken with a lazy consensus approach: a few positive votes with no negative vote is enough to get going.

Voting is done with numbers:

- +1 -- a positive vote
- 0 -- abstain, have no opinion
- -1 -- a negative vote

The rules require that a negative vote includes an alternative proposal or a detailed explanation of the reasons for the negative vote.

The community then tries to gather consensus on an alternative proposal that resolves the issue. In the great majority of cases, the concerns leading to the negative vote can be addressed.

This process is called "consensus gathering" and we consider it a very important indication of a healthy community.

Specific cases have some more detailed voting rules.

http://www.apache.org/foundation/how-it-works.html