Part VI

Génie Logiciel Libre

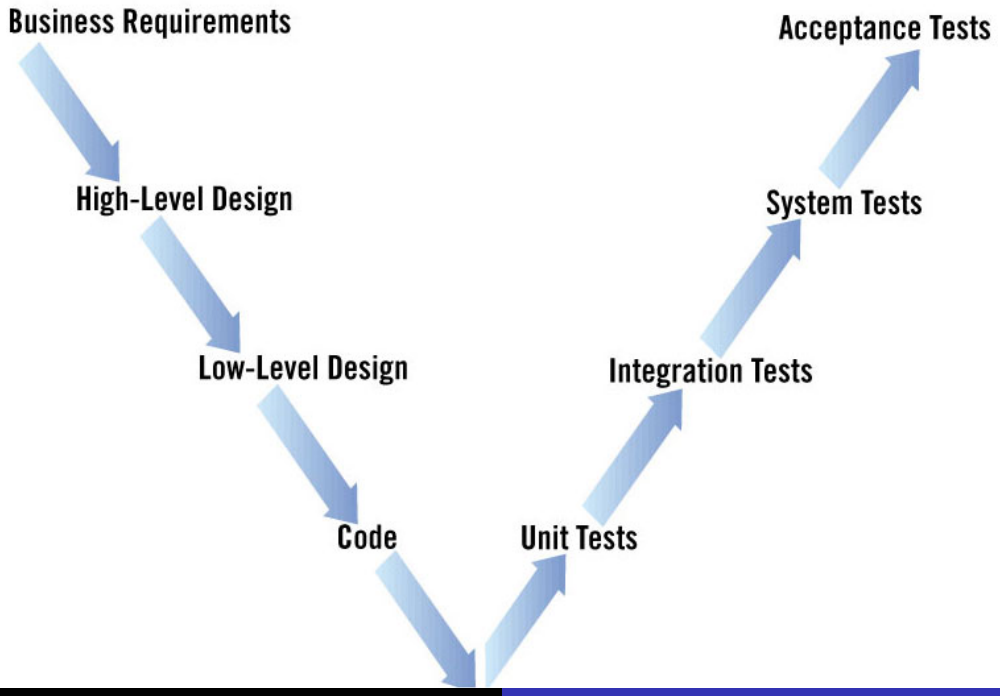
16 Génie Logiciel
- Linux
- Eric S. Raymond
- Apache
- Mythes et réalité

17 Securité
- Exemples du monde propriétaire
- Vote éléctronique

18 Distributions Linux
- Packages et dependances
- Outils existants
- Installabilité et SAT
- Un exemple

Un peu d'histoire du Génie logiciel

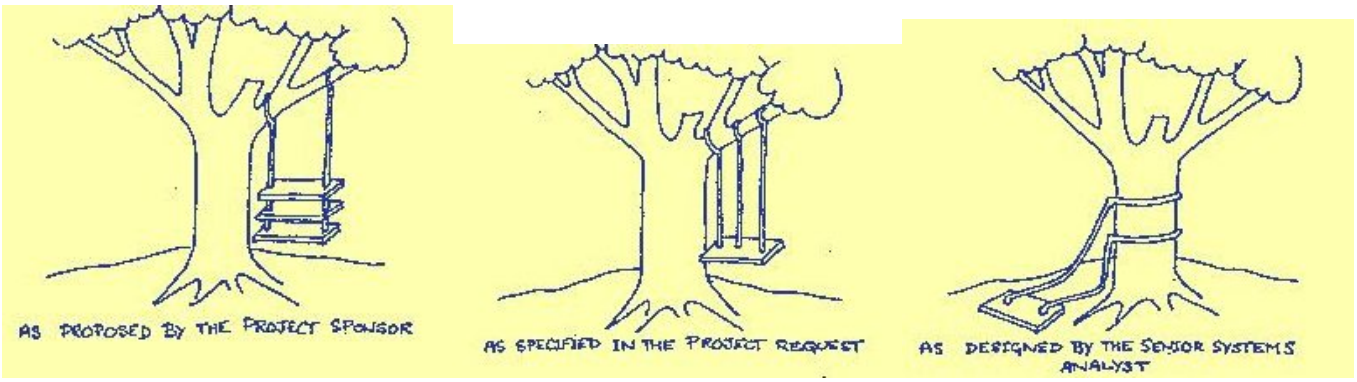
Le "modèle à V" incarne la vision du dévéloppement logiciel typique des années '80



Défauts du modèle

- il faut une organisation centralisée
- les spécifications initiales peuvent être fausses ou incomplètes
- la distance entre spécifications et test final peut être trop longue
- l'utilisateur final est exclu du processus jusqu'à la fin
- quand on découvre un erreur grave, il est souvent trop tard

Le problème en six images



Des solutions proposées ensuite. . .

- programmation orientée objet
- conception et modélisation via UML
- utilisation de méthodes formelles dans le dévéloppement
- diverses méthodologies plus récentes:
  - extreme programming
  - méthodes agiles
  - . . .

Voyons voir concrètement ce qui s'est passé avec certains projets en logiciel libre.

Analyses

Exemples de communautés significatives

- Linux
- Apache

Linux: une radiographie

- quelques dates:
  vers. 0.02 en 10/1991; 0.95 en 03/1992; 1.0 en 03/1994
- modèle du *dictateur bienveillant*:
  Linus Torvalds a le dernier mot sur tout
- élaboration des patchs sur la mailing list (cela permet de discuter les propositions)
- séparation entre fils de versions:
  stable (2.4.x, 2.6.x), et *experimental* (1.3.xx or 2.1.x)
- gestion de version avancée (plus de détails dans un cours à part)

Exemple de patch dans la mailing list

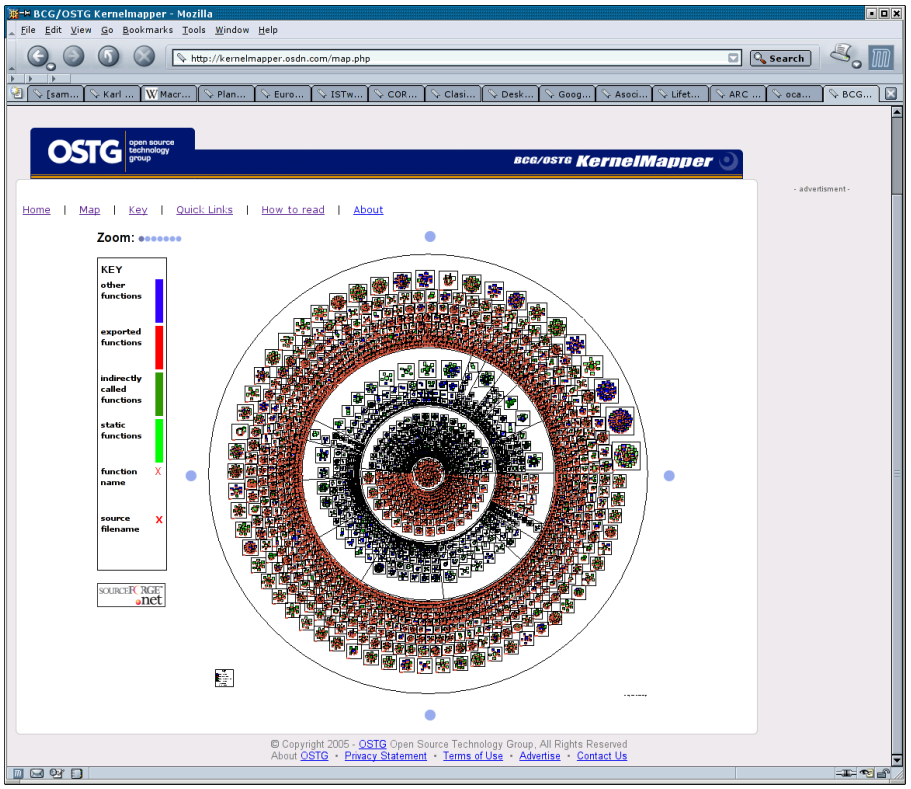
```
Date Fri, 16 Mar 2007 12:36:15 +0100
From Jarek Poplawski <>
Subject [PATCH] Re: Fwd: oprofile lockdep warning on rc1

On 28-02-2007 01:46, Dave Jones wrote:
> This happened on a 2.6.21rc1 kernel.
...
> Richard Hughes <hughsient@gmail.com>
> Date:
> Tue, 27 Feb 2007 21:59:07 +0000
> To:
> Development discussions related to Fedora Core
> <fedora-devel-list@redhat.com>
...
> But also I get this (!!!):
>

Here is my patch proposal for testing.

Regards,
Jarek P.
```

Roberto Di Cosmo
Logiciel libre, une introduction
Génie Logiciel
Securité
Distributions Linux
Linux
Eric S. Raymond
Apache
Mythes et réalité

Exemple de patch dans la mailing list

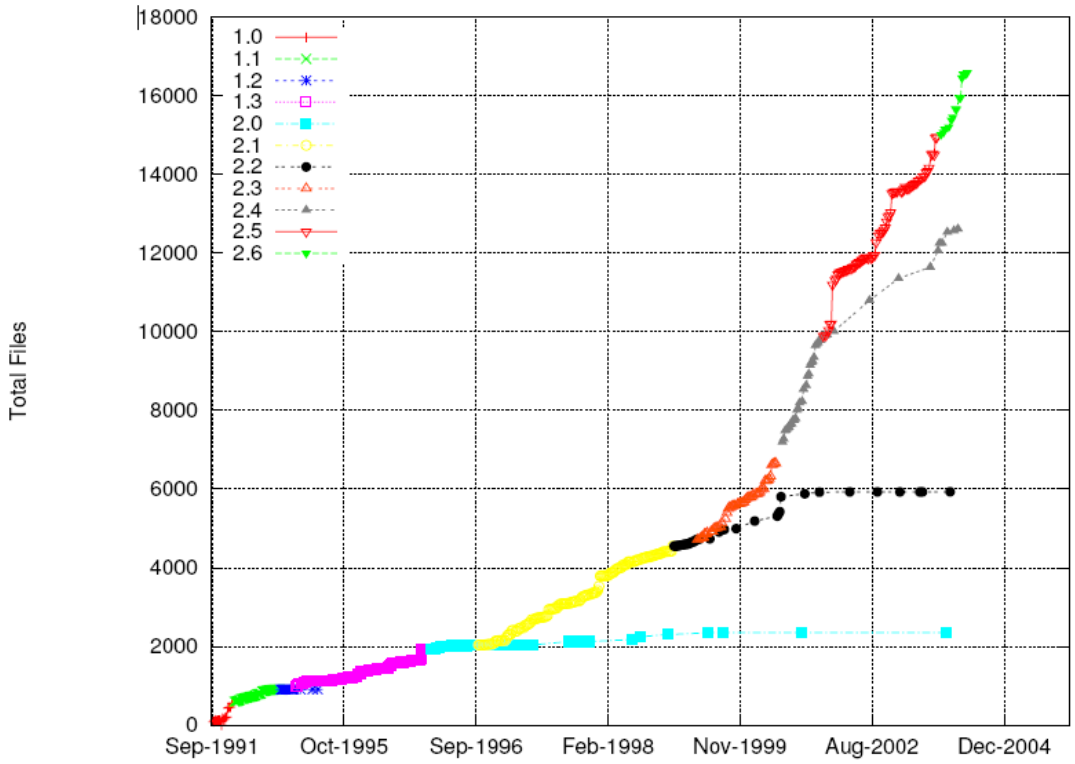
```
lockdep found oprofilefs_lock is taken both in process
context (oprofilefs_ulong_from_user()) and from hardirq
(nmi_cpu_setup()), so the lockup is possible.

Reported-by: Richard Hughes <hughsient@gmail.com>
Signed-off-by: Jarek Poplawski <jarkao2@o2.pl>

---
diff -Nurp linux-2.6.21-rc1-/drivers/oprofile/oprofilefs.c linux-2.6.21-rc1/drivers/oprofile/oprofilefs.c
--- linux-2.6.21-rc1-/drivers/oprofile/oprofilefs.c 2007-02-27 10:47:38.000000000 +0100
+++ linux-2.6.21-rc1/drivers/oprofile/oprofilefs.c 2007-03-16 11:36:30.000000000 +0100
@@ -65,6 +65,7 @@ ssize_t oprofilefs_ulong_to_user(unsigned
 int oprofilefs_ulong_from_user(unsigned long * val, char const __user * buf, size_t count)

  char tmpbuf[TMPBUFSIZE];
+ unsigned long flags;

  if (!count)
  return 0;
@@ -77,9 +78,9 @@ int oprofilefs_ulong_from_user(unsigned
  if (copy_from_user(tmpbuf, buf, count))
  return -EFAULT;

- spin_lock(&oprofilefs_lock);
+ spin_lock_irqsave(&oprofilefs_lock, flags);
  *val = simple_strtoul(tmpbuf, NULL, 0);
- spin_unlock(&oprofilefs_lock);
+ spin_unlock_irqrestore(&oprofilefs_lock, flags);
```

Un aperçu: structuration



Génie Logiciel
Securité
Distributions Linux
Linux
Eric S. Raymond
Apache
Mythes et réalité

Un aperçu: évolution



Analyses

Un texte très connu tire des leçons du succès de Linux :

The cathedral and the Bazaar, Eric S. Raymond, 1997

cathedral : processus logiciel lourd et hierarchique

bazaar : dévéloppement par des équipes avec faible couplage

Règles de bonne conduite énoncées

1. Every good work of software starts by scratching a developer's personal itch.
2. Good programmers know what to write. Great ones know what to rewrite (and reuse).
3. "Plan to throw one away; you will, anyhow." (Fred Brooks, The Mythical Man-Month, Chapter 11)
4. If you have the right attitude, interesting problems will find you.
5. When you lose interest in a program, your last duty to it is to hand it off to a competent successor.
6. Treat your users as co-developers.
7. Release early. Release often.
8. Given enough eyeballs, all bugs are shallow.

Critiques

- analyse très orientée Linux
- separation entre Bazaar et Cathedral moins nette dans la réalité
- beaucoup d'approximations, par exemple, la dénégation de la loi de Brooks, qui oublie la différence entre core developers et utilis-acteurs à la marge du projet

Apache

- quelques dates:
  Création du projet en 1995 par Rob McCool, première version stable en 1996, version 1.3 en 1998, IBM choisit Apache pour WebSphere en juin 1998
- approx. 1500 committers sur les projets (une centaine sur httpd)
- organisation *oligarchique*

Committers get a shot at working on the code; good committers become members and thus get a piece of the ownership of the software and the direction. Commit access is a privilege, not a right, and is based on trust. The Apache Software Foundation is a meritocracy [. . .]
New candidates for membership are nominated by an existing member and then put to vote; a majority of the existing membership must approve a candidate in order to the candidate to be accepted.

Quelques mythes sur le Logiciel Libre

- dévéloppement anarchique sans leader
- connaissance qui emerge naturellement de la sagesse des masses

  The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations
  2004 James Surowiecki

- le logiciel libre est *toujours* de meilleure qualité que le logiciel propriétaire
- l'écosystème du logiciel libre est basé sur des valeurs d'altruisme communautaire...

Reverse Software Engineeering

On peut essayer d'analyser ces cas *systématiquement*:

- disponibilité des sources
- accès aux données historiques
- informations complètes sur les dévéloppeurs
- pas besoin d'accords formels

Un travail d'envergure a été entrepris par
`http://libresoft.urjc.es/index`

Un point de vue moderne

Les analyses de Martin Michlmayr (ex Debian project leader) sont disponibles sur `http://opensource.mit.edu`

Cathedral phase	Transition phase	Bazaar phase
Original "idea" Project Author Core developers Unix philosophy	⇒ "Interest" Prototype Modular design	⇒ Distributed development environment Community Parallel perfective and corrective maintenance Peer reviews

Sécurité et Logiciel libre

- les modèles de sécurité
- l'exemple du vote éléctronique

Modèles de sécurité

Fermé On construit un système dont les spécifications sont secrètes.
La sécurité repose en partie sur le secret des spécifications
(Ex: Enigma).

Avantages:

- On ajoute une difficulté supplémentaire pour les casseurs

Desavantages:

- La difficulté supplémentaire donne une fausse assurance
- Modèle traditionnellement sensible à la traison
- Difficile, voir impossible à mettre en oeuvre avec des composantes disponibles sur le marché

Modèles de sécurité

Difficile, voir impossible à mettre en oeuvre avec des composantes disponibles sur le marché,...

pourquoi?

- on ne maîtrise pas le code source propriétaire
- on ne sait pas faire d'audit de sécurité complet automatique sur 50M lignes de code
- chacune de ces lignes de code est suspecte
- même les outils de compilation[32] sont suspects!

[32] Ken Thompson: Reflections on trusting trust

Reflections on trusting trust

Modèles de sécurité

Ouvert les spécifications du système sont connues de tous. La sécurité repose en partie sur la vérification par les paires (Ex: RSA).

Avantages:

- Modèle étanche à la traison (inutile de voler un decodeur RSA).
- Modèle facile á mettre en place avec des composantes du marché, à condition qu'elles soient aussi "ouvertes" (i.e. livrées avec leur code)

Desavantages:

- La qualité de la protection depende seulement de la qualité des algorithmes et de leur mise en oeuvre (crypto encore une art).
- La qualité de la protection depende aussi de la rapidité de correction des erreurs de conception ou mise en oeuvre