[Ksummit-discuss] [MAINTAINER TOPIC] Succession Planning: Is It time to Throw Linus Under a Bus?

Daniel Vetter [daniel.vetter at ffwll.ch](#)
Sat Sep 8 12:21:19 UTC 2018

- Previous message: [[Ksummit-discuss] [MAINTAINER TOPIC] Succession Planning: Is It time to Throw Linus Under a Bus?](#)
- Next message: [[Ksummit-discuss] [MAINTAINER TOPIC] Succession Planning: Is It time to Throw Linus Under a Bus?](#)
- **Messages sorted by:** [\[date \]](#) [\[thread \]](#) [\[subject \]](#) [\[author \]](#)

```
On Sat, Sep 8, 2018 at 12:47 PM, Mauro Carvalho Chehab
<mchehab+samsung at kernel.org> wrote:
> Em Thu, 06 Sep 2018 12:51:15 -0700
> James Bottomley <James.Bottomley at HansenPartnership.com> escreveu:
>
>> On Thu, 2018-09-06 at 21:47 +0200, Daniel Vetter wrote:
>> > On Thu, Sep 6, 2018 at 9:44 PM, James Bottomley
>> > <James.Bottomley at hansenpartnership.com> wrote:
>> > > Since our fearless leader apparently can't even remember the dates
>> > > of
>> > > the only conference he goes to, perhaps now might be a good time to
>> > > talk about how we'd run an orderly succession process (purely
>> > > theoretically, of course ...)
>> > >
>> > > I think a vote amongst the Maintainer Summit attendees might be the
>> > > way
>> > > to elect a new leader, but others probably have different ideas.
>> >
>> > Why I single leader?
>> >
>> > This group maintainer ship thing ... it  works.
>>
>> I also note that group maintainership seems only to work for you in
>> DRM; most of the other subsystems seem to have single leader
>> hierarchical maintainership.
>
> If we're seriously thinking on have a plan for maintainership
> replacement with a group, we need first to answer:
>
>         Q: How many Kernel developers does it take to replace Linus?
>
> That's said, I doubt that the DRM model would work outside DRM
> subsystem: too many people with commit rights can be hard, specially
> if they all can touch the core. Btw, we tried a similar model in
> the past on media (by the time we were using Mercurial, about
> 10 years ago): all core developers had commit rights. It worked
> smoothly for years, until one of the developers decided
> to commit a very intrusive patch touching all drivers at the
> subsystem and stepping on everyone else feet. Handling it was
> really painful. If our official model would be a group
```

> *maintainership, it would probably take months for us to put the*
> *tree on a sane state. I ended by using my maintainership status*
> *to revert the patch, returning the tree to a sane state and allowing*
> *the others to keep sending patches. Yet, it took months, even*
> *years for us to recover from the disaster. Also, due to the*
> *heated discussions, we end by losing several developers due*
> *to that.*

I'm not advocating for a committer model at the top level, but it
would be awesome if people bothered to look at the talks and blogs
we've done about what we're actually doing with commit rights, instead
of making assumptions. We don't just throw out commit rights like hot
candy, since that indeed is bound to fail. Instead:

- We have clearly documented merge criteria, and as much of that as
possible enforced using tooling.
- One of those is mandatory review, no one is allowed to do anything solo.
- We have massive CI, available to all contributors automatically –
that gives us mandatory in-depth testing way before committing is even
on the table.
- And finally we have a CoC to just ban people who don't get it and
cant work together in a group
- a bunch of smaller things all over to make it fit together

I know that the commit right thing is very radical by kernel
standards. But outside of the kernel, "how to make commit rights work"
is largely a solved problem. Of course if you ignore that trove of
experience from other open source projects, then you're bound to
repeat all the fail, no surprises.

Apologies for the interlude, back to the topic (which is apparently
... throwing Linus under the bus?!?).
–Daniel

> *A triumvirate like x86 and arm might work, but I suspect that it*
> *would be easier to start with a more hierarchical model, like*
> *for example having one (sub-)maintainer for arch, another*
> *one for core and a third one for drivers.*
>
> *Also, if we want to consider the bus scenario and other disaster*
> *threats, it could be worth to consider having them geographically*
> *distributed (if possible even on different Countries).*
>
> *Thanks,*
> *Mauro*

--
Daniel Vetter
Software Engineer, Intel Corporation
+41 (0) 79 365 57 48 – http://blog.ffwll.ch

[More information about the Ksummit-discuss mailing list](#)