Maintainers Don't Scale

Daniel Vetter, Intel OTC
Kernel Recipes 2016, Paris

why?

- impossible to volunteer new co-maintainers
- perception of bottleneck in running

```
$ git apply-mbox
```

why not … ?

- submaintainers or topic trees (arm-soc model)
- moar scripts! (gregkh model)

(after) KS 2015

- Linus endorsing group maintainer models
- 2 maintainers, ~15 committers
- … it's just a big group

disaster failed to materialize

- over the top happy contributors
- much better mesh collaboration
- now suddenly a bored maintainer
- commit rights is a really good carrot
- bonus: distributed integration conflict handling

maintainer as a service

- coordination with other trees
- consensus engineering
- connecting people
- pull requests & nonlinear commits
- taking the blame

ingredients: a team

- non-maintainer reviews as the norm
- consistent group
- drive-by contributions as a minority

ingredients: docs&tools

- dim maintainer tools:
  https://cgit.freedesktop.org/drm/drm-intel/log/?h=maintainer-tools
- process documentation:
  https://01.org/linuxgraphics/gfx-docs/maintainer-tools/drm-intel.html
- manpage, bash completion & kitchen sink

ingredients: CI

- IGT gpu tools&tests: https://cgit.freedesktop.org/drm/igt-gpu-tools/
- series-aware patchwork to tie into m-l
- hard problem

ingredients: rough consensus

- defaulting to no action
- agreement on goals and target
- disagreement on the path is acceptable
- stakeholders = committers/regular contributors

ingredients: bugfix flow

- all bugfixes applied to -next
- only maintainers cherry-pick to -fixes
- `git merge -X ours` for backmerges
- handled using Fixes: tags

next (well, at KS)

- updating `MAINTAINERS`
- tree-wide refactorings vs. bugfix flow
- more maintainer teams like this?
- KS invitations?

summary

- much more scalable leaf maintainer model
- (big) group resilience
- integrates into the maintainer hierarchy
- now rolling out for drm-misc