Linux Kernel Development

Greg Kroah-Hartman
gregkh@linuxfoundation.org

github.com/gregkh/kernel-development

58,000 files
23,100,000 lines

Kernel release 4.11.0

3,781 developers
≈400 companies

Kernel releases 4.7.0 – 4.11.0
May 2016 – April 2017

7,300 lines added
2,400 lines removed
2,000 lines modified

Kernel releases 4.7.0 – 4.11.0
May 2016 – April 2017

7,300 lines added
2,400 lines removed
2,000 lines modified

Every day

Kernel releases 4.7.0 – 4.11.0
May 2016 – April 2017

8 changes per hour

Kernel releases 4.7.0 – 4.11.0
May 2016 – April 2017

9.7 changes per hour

4.9 release

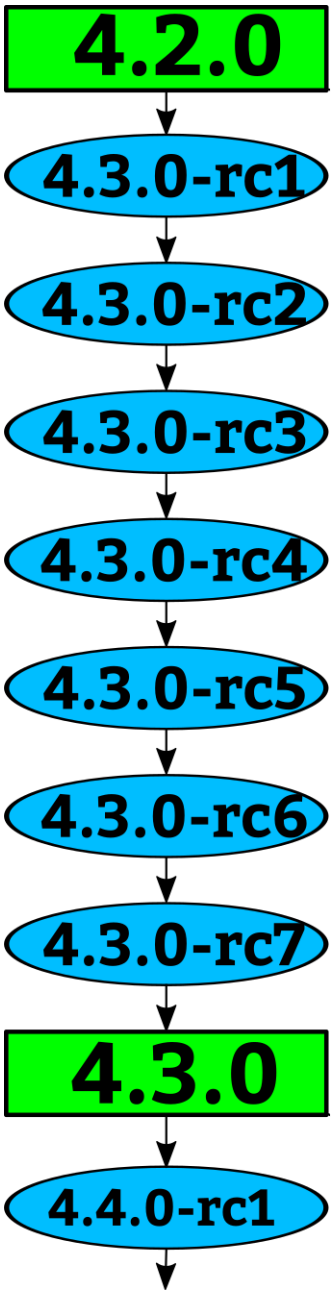
4.12 release July 7th?

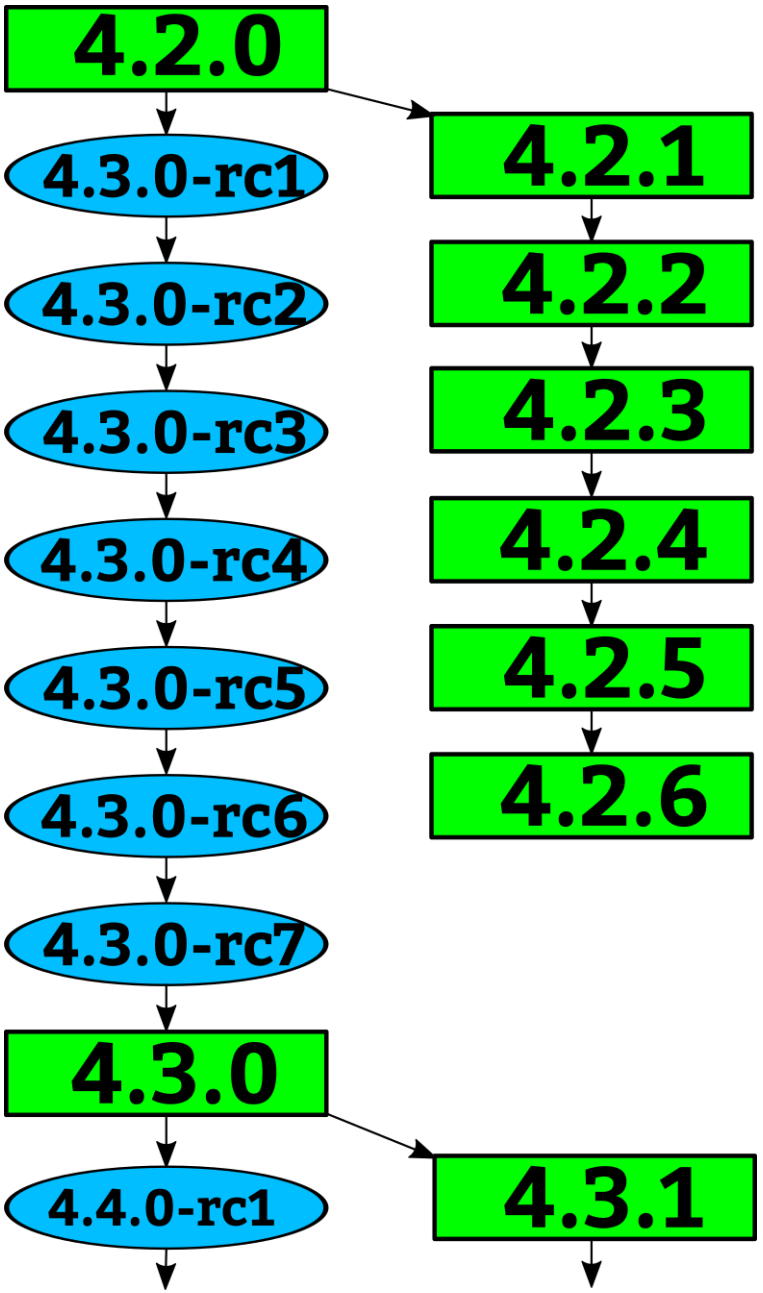
2nd largest release

How we stay sane

Time based releases
Incremental changes

Kernel releases 3.10.0 – 3.14.0


4.2.0
4.3.0-rc1
4.3.0-rc2
4.3.0-rc3
4.3.0-rc4
4.3.0-rc5
4.3.0-rc6
4.3.0-rc7
4.3.0
4.4.0-rc1

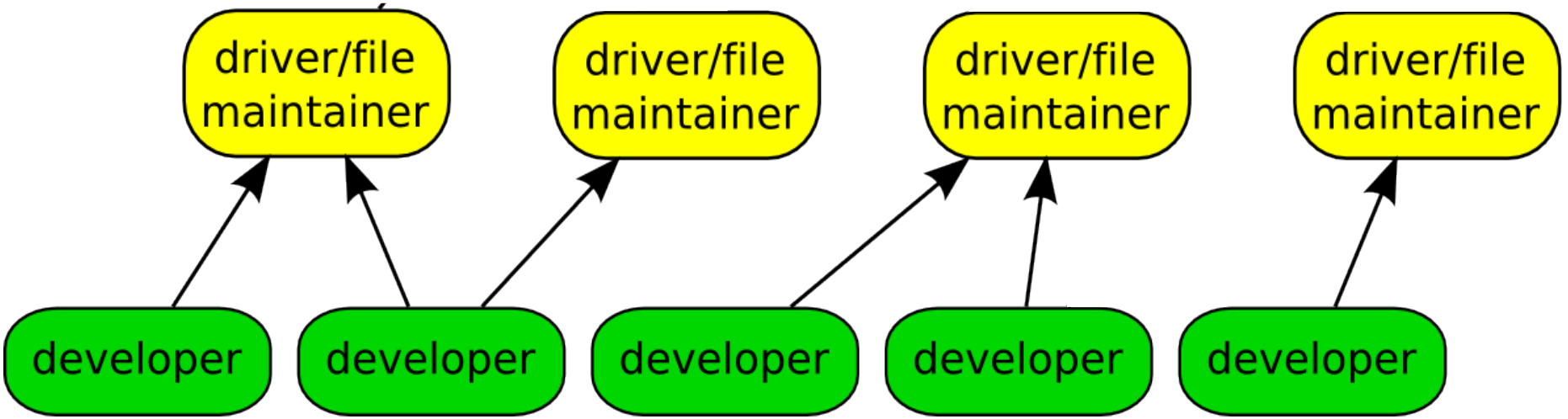

4.2.0
4.3.0-rc1
4.3.0-rc2
4.3.0-rc3
4.3.0-rc4
4.3.0-rc5
4.3.0-rc6
4.3.0-rc7
4.3.0
4.4.0-rc1
4.2.1
4.2.2
4.2.3
4.2.4
4.2.5
4.2.6
4.3.1

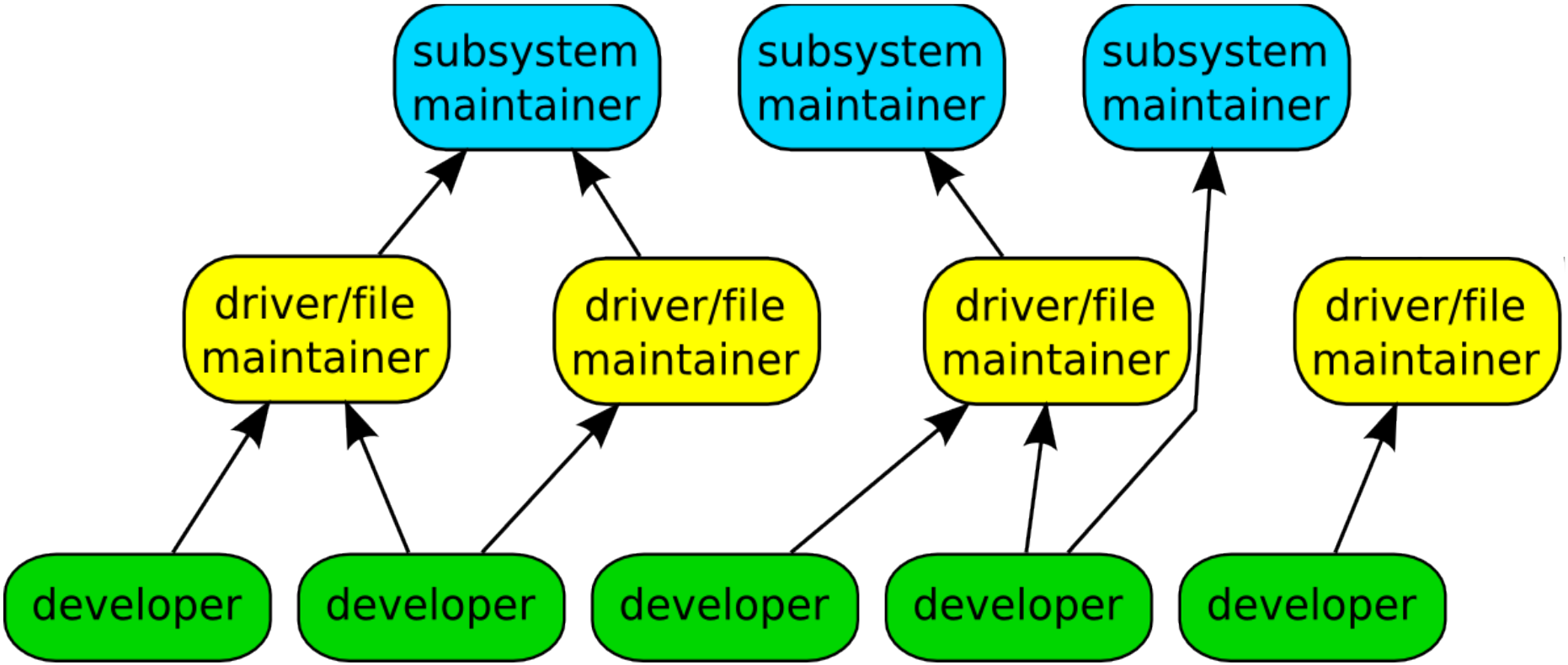
“Longterm kernels”

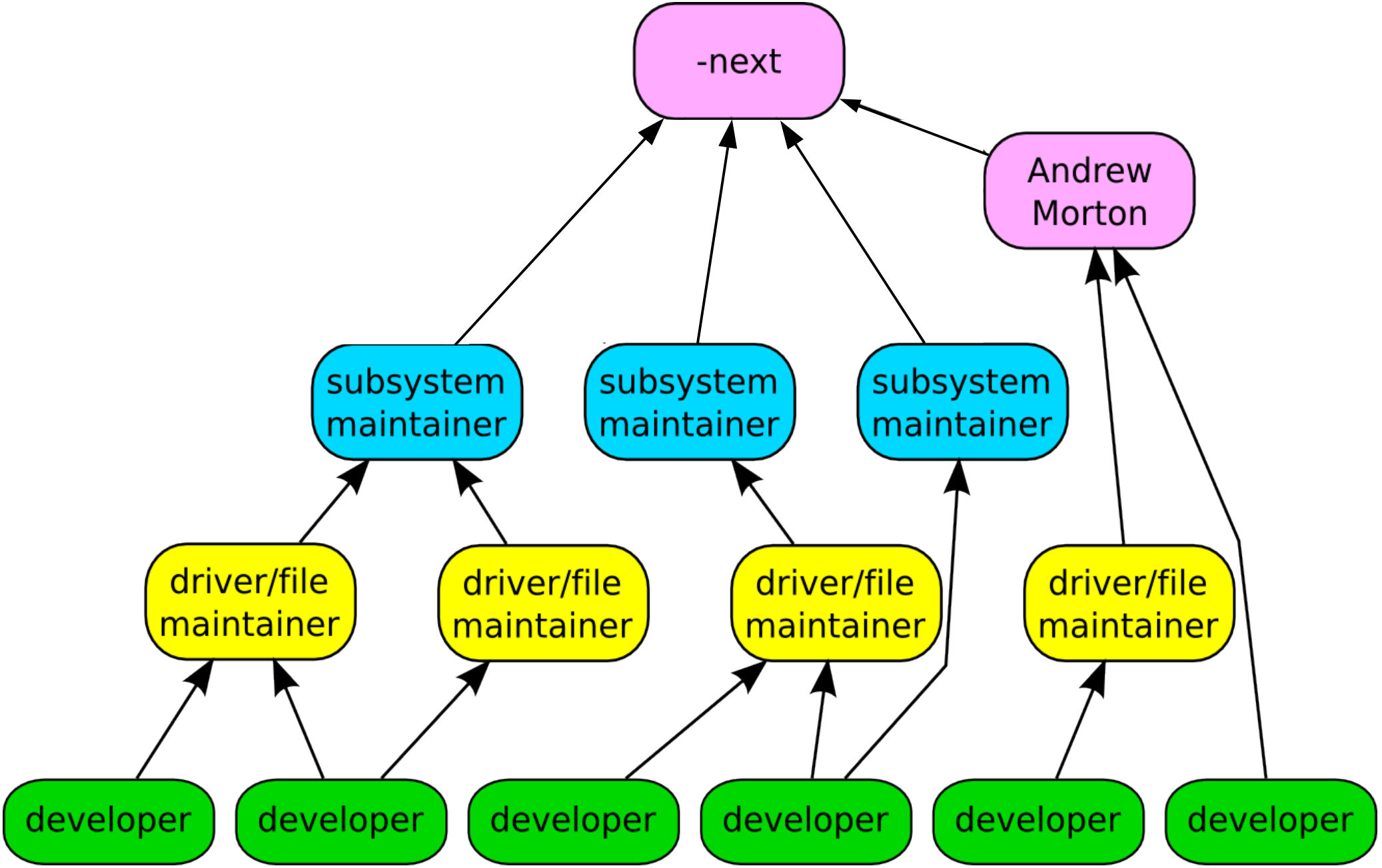
One picked per year
Maintained for two years

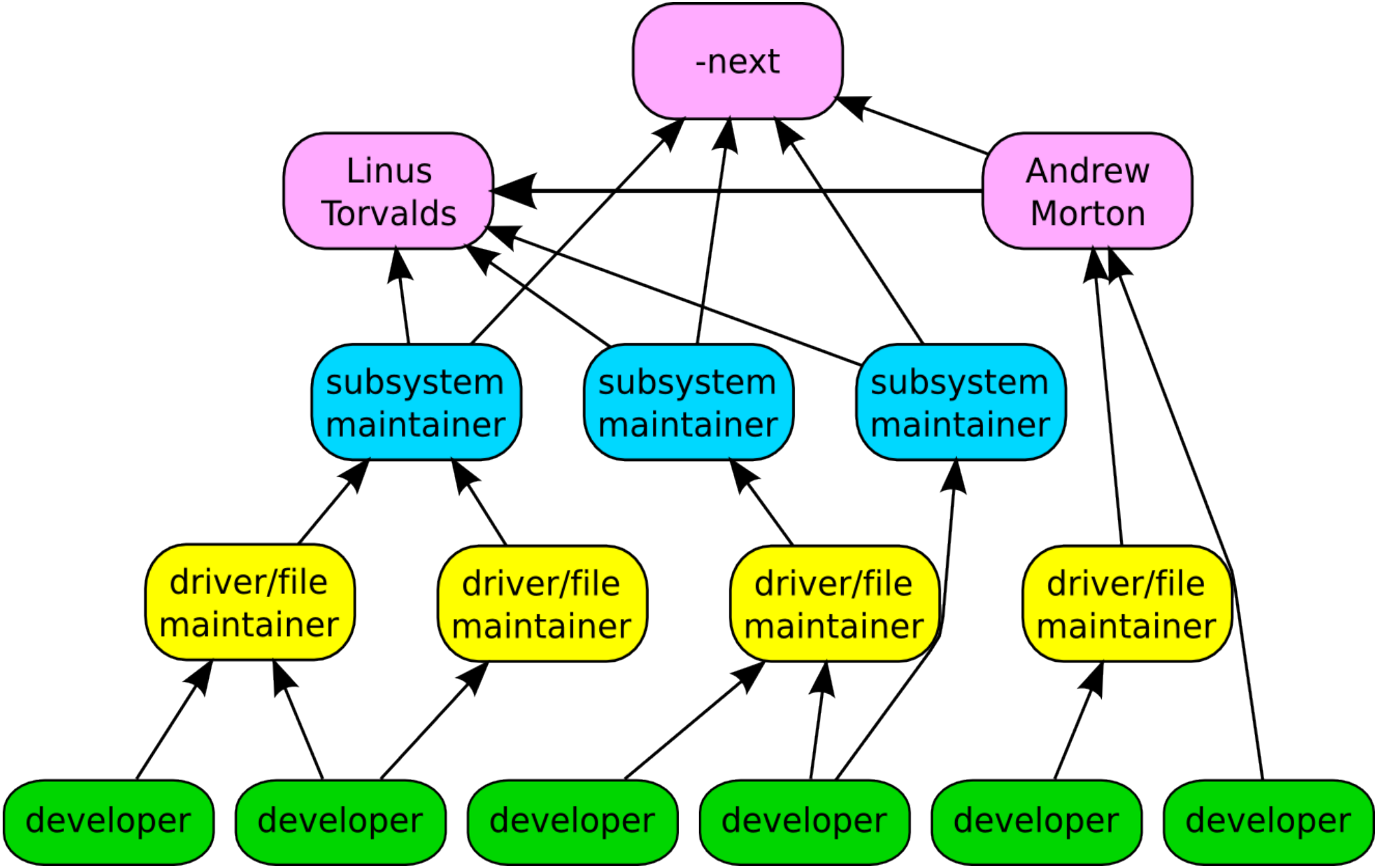
4.4 4.9


developer
developer
developer
developer
developer


driver/file maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
developer
developer
developer
developer
developer


subsystem maintainer
subsystem maintainer
subsystem maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
developer
developer
developer
developer
developer


-next
Andrew Morton
subsystem maintainer
subsystem maintainer
subsystem maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
developer
developer
developer
developer
developer
developer


-next
Linus Torvalds
Andrew Morton
subsystem maintainer
subsystem maintainer
subsystem maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
developer
developer
developer
developer
developer
developer

Top developers by quantity

Chris Wilson	1093
Mauro Carvalho Chehab	845
Johan Hovold	754
Arnd Bergmann	714
Viresh Kumar	537
Geert Uytterhoeven	473
Christoph Hellwig	456
Wei Yongjun	451
Ville Syrjälä	446
Linus Walleij	411
Greg Kroah-Hartman	409

Kernel releases 4.7.0 – 4.11.0

Top Signed-off-by:

Greg Kroah-Hartman	7734
David S. Miller	7107
Mauro Carvalho Chehab	2317
Linus Torvalds	2144
Mark Brown	1966
Andrew Morton	1930
Ingo Molnar	1809
Alex Deucher	1529
Linus Walleij	1202
Chris Wilson	1199
Kalle Valo	1196

Kernel releases 4.7.0 – 4.11.0

Who is funding this work?

1. “Amateurs”	14.4%
2. Intel	13.4%
3. Red Hat	7.3%
4. Linaro	6.4%
5. IBM	3.4%
6. Samsung	3.4%
7. Consultants	3.0%
8. SuSE	2.9%
9. Google	2.7%
10. AMD	2.3%

Kernel releases 4.7 – 4.11

Who is funding this work?

11. Mellanox	1.9%
12. Renesas Electronics	1.8%
13. Huawei Technologies	1.6%
14. Oracle	1.6%
15. Broadcom	1.5%
16. Texas Instruments	1.5%
17. ARM	1.4%
18. Free Electrons	1.1%
19. Imagination Technologies	1.0%
20. NXP Semiconductors	0.9%

Kernel releases 4.7 – 4.11

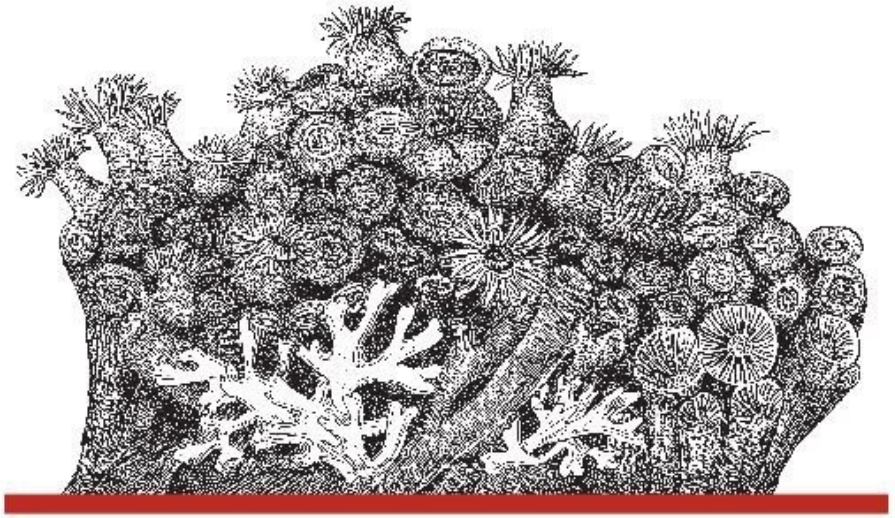
“Working upstream saves time and money”

Dan Frye – VP Open Systems, IBM
Dirk Hohndel – Chief Technologist, Intel

Getting involved

Run the kernel.org release on your machine

Getting involved



LINUX KERNEL

IN A NUTSHELL

A Desktop Quick Reference

Getting involved

`Documentation/HOWTO`

`Documentation/development-process`

Getting involved

kernelnewbies.org

Getting involved

Google “write your first kernel patch”

Getting involved

kernelnewbies.org/KernelJanitors/Todo

Getting involved

Linux Driver Project

`drivers/staging/*/TODO`

Getting involved

Eudyptula Challenge
(little penguin)

http://eudyptula-challenge.org/

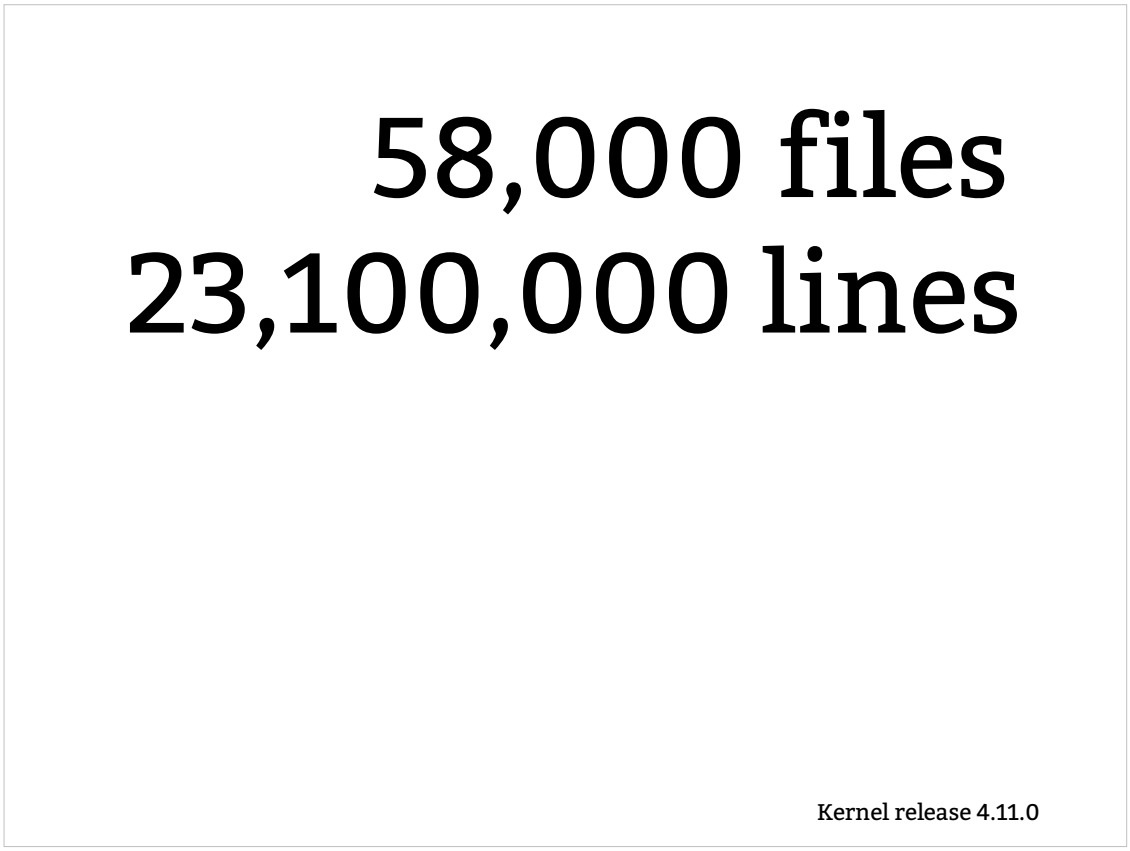
github.com/gregkh/kernel-development

Linux Kernel Development

Greg Kroah-Hartman
gregkh@linuxfoundation.org

github.com/gregkh/kernel-development

I'm going to discuss the how fast the kernel is moving, how we do it all, and how you can get involved.



This was for the 4.11 kernel release, which happened April 30, 2017.

3,781 developers
≈400 companies

Kernel releases 4.7.0 – 4.11.0
May 2016 – April 2017

This makes the Linux kernel the largest contributed body of software out there that we know of.

This is just the number of companies that we know about, there are more that we do not, and as the responses to our inquiries come in, this number will go up.

Have surpassed 400 companies for 4 years now.

7,300 lines added
2,400 lines removed
2,000 lines modified

Kernel releases 4.7.0 – 4.11.0
May 2016 – April 2017

7,300 lines added
2,400 lines removed
2,000 lines modified

Every day

Kernel releases 4.7.0 – 4.11.0
May 2016 – April 2017

8 changes per hour

Kernel releases 4.7.0 – 4.11.0
May 2016 – April 2017

This is 24 hours a day, 7 days a week, for a full year.

We went this fast the year before this as well, this is an amazing rate of change.

Interesting note, all of these changes are all through the whole kernel.

For example, the core kernel is only 5% of the code, and 5% of the change was to the core kernel. Drivers are 55%, and 55% was done to them, it's completely proportional all across the whole kernel.

9.7 changes per hour

4.9 release

4.9 was the “largest” in number of changes that we have ever accepted. After 4.9, things went down a bit for 4.10 and 4.11, but 4.12 is getting very big.

Now this is just the patches we accepted, not all of the patches that have been submitted, lots of patches are rejected, as anyone who has ever tried to submit a patch can attest to.

4.12 release July 7th?

2nd largest release

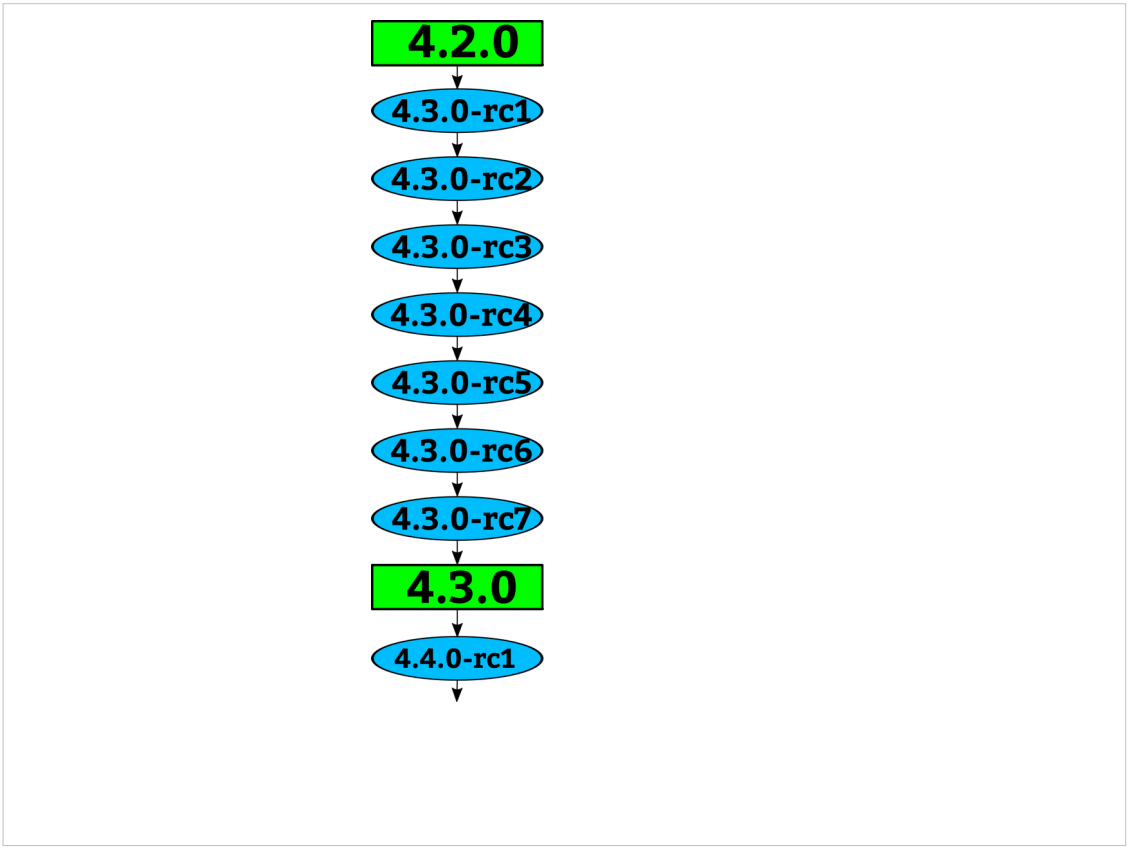
4.12 should be released on July 7th and is on track to be the 2nd largest release by number of changes we have ever done.

And the first largest on number of lines of code we have added, due to some very large drivers being added to the tree.

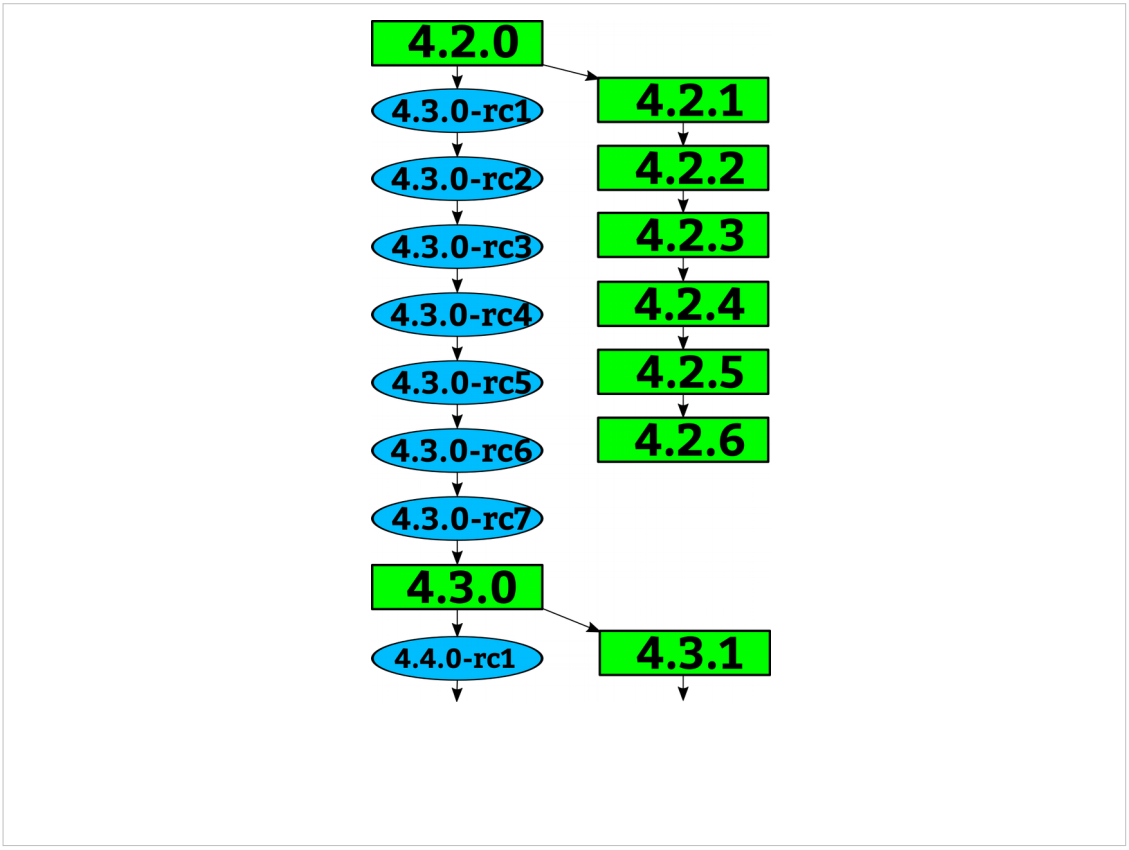
How we stay sane

Time based releases
Incremental changes

67 days to be exact, very regular experience.



How a kernel is developed.
Linus releases a stable kernel
- 2 week merge window from subsystem maintainers
- rc1 is released
- bugfixes only now
- 2 weeks later, rc2
- bugfixes and regressions
- 2 weeks later,rc3
And so on until all major bugfixes and regressions are resolved and then the cycle starts over again.



Greg takes the stable releases from Linus, and does stable releases with them, applying only fixes that are already in Linus's tree.

Requiring fixes to be in Linus's tree first ensures that there is no divergence in the development model.

After Linus releases a new stable release, the old stable series is dropped.

With the exception of “longterm” stable releases, those are special, the stick around for much longer...

“Longterm kernels”

One picked per year
Maintained for two years

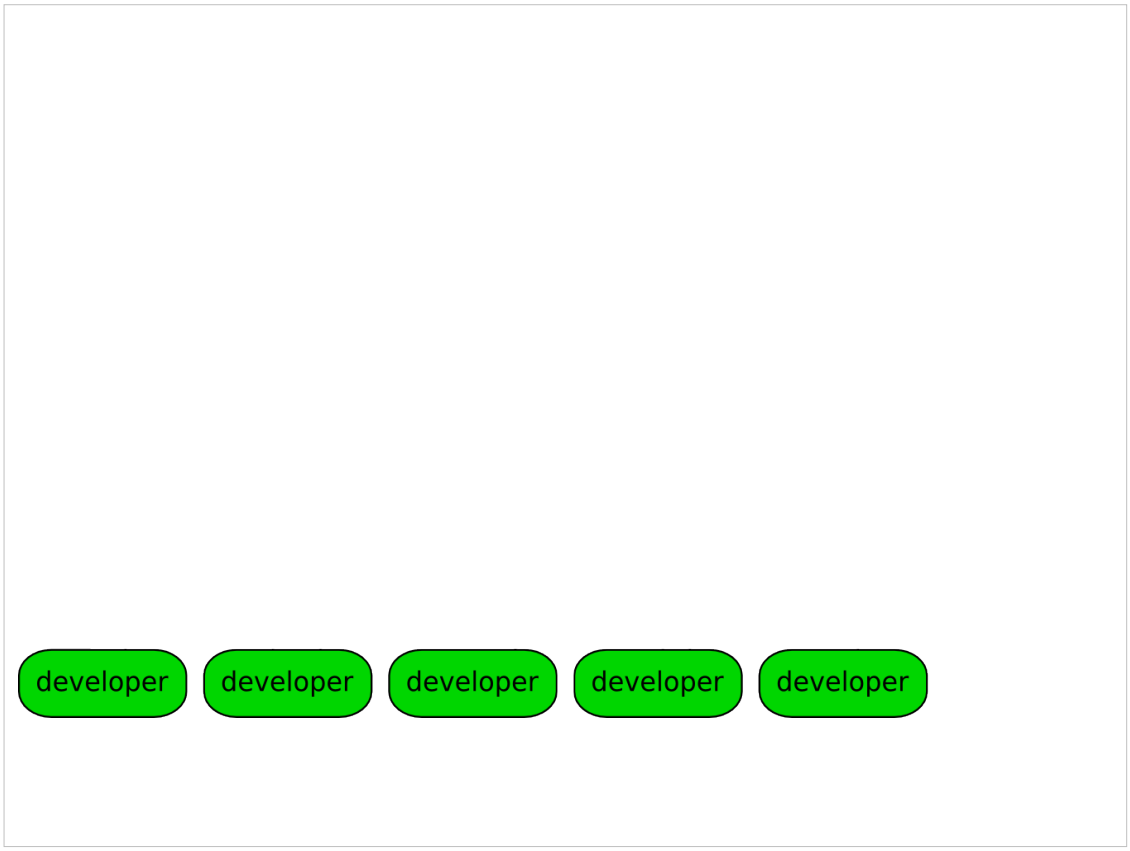
4.4 4.9

I pick one kernel release per year to maintain for longer than one release cycle. This kernel I will maintain for at least 2 years.

This means there are 2 longterm kernels being maintained at the same time.

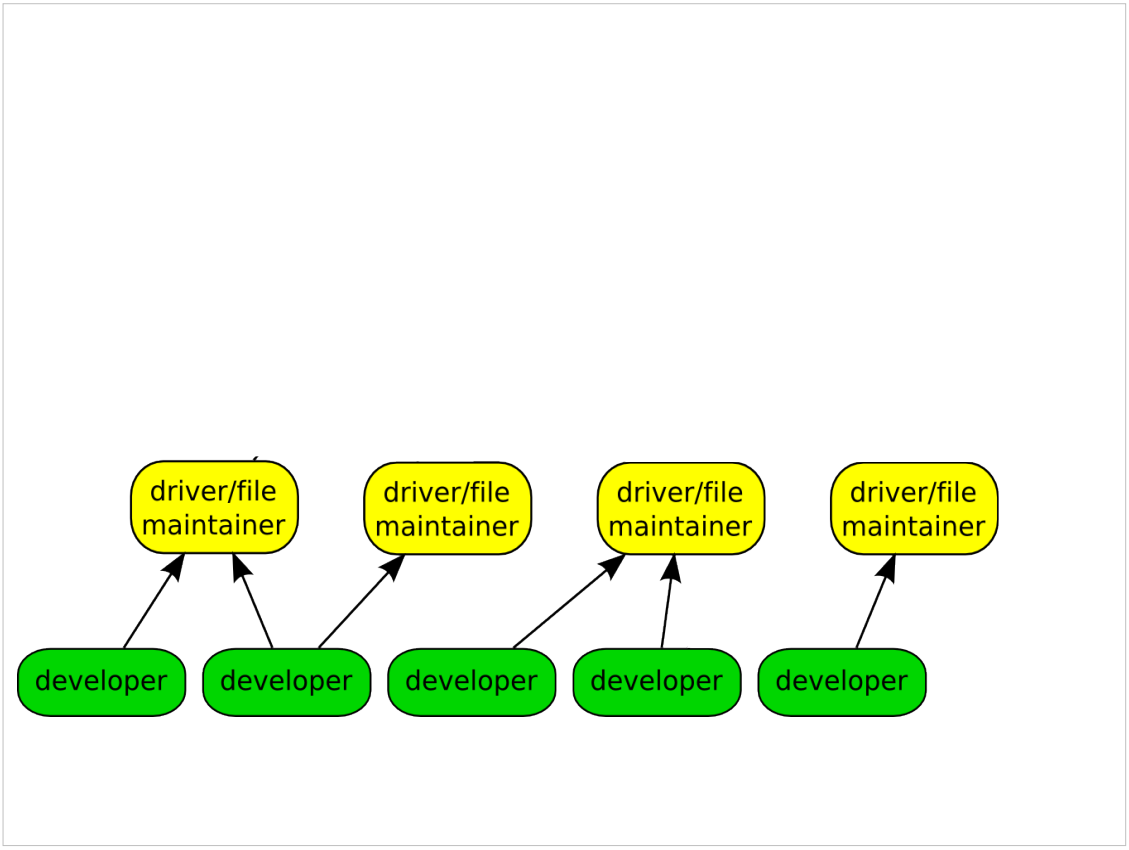
4.4 and 4.9 are the longterm kernel releases I am currently maintaining

The LTSI project is based on the longterm kernels.



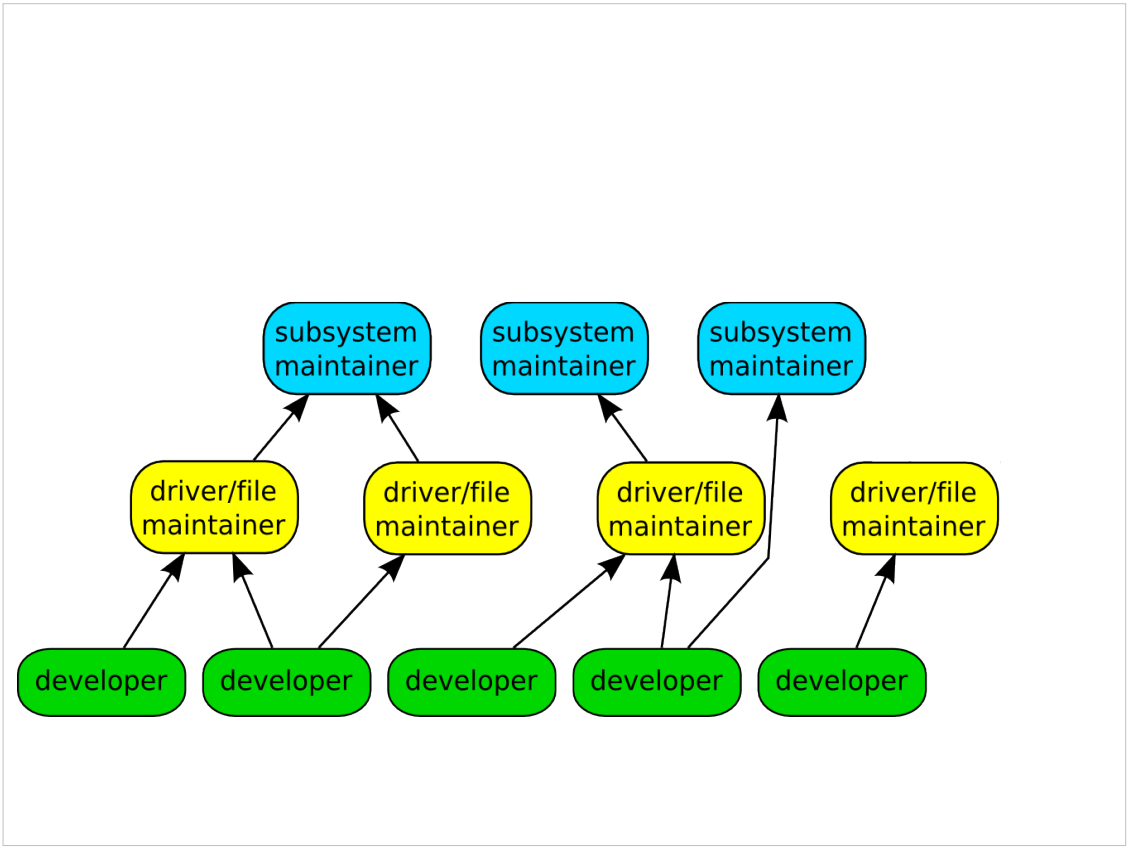
Like mentioned before, we have almost 3000 individual contributors. They all create a patch, a single change to the Linux kernel. This change could be something small, like a spelling correction, or something larger, like a whole new driver.

Every patch that is created only does one thing, and it can not break the build, complex changes to the kernel get broken up into smaller pieces.



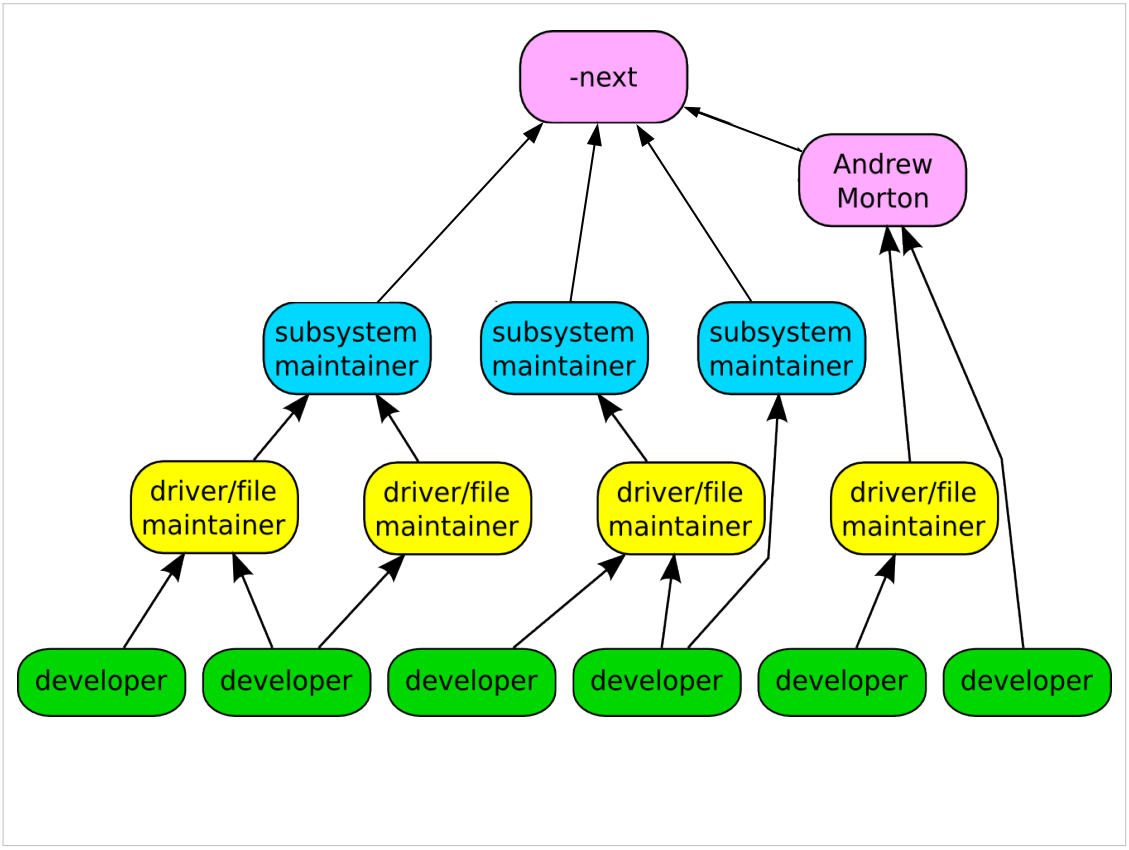
The developers send their patch to the maintainer of the file(s) that they have modified.

We have about 700 different driver/file/subsystem maintainers



After reviewing the code, and adding their own signed-off-by to the patch, the file/driver maintainer sends the patch to the subsystem maintainer responsible for that portion of the kernel.

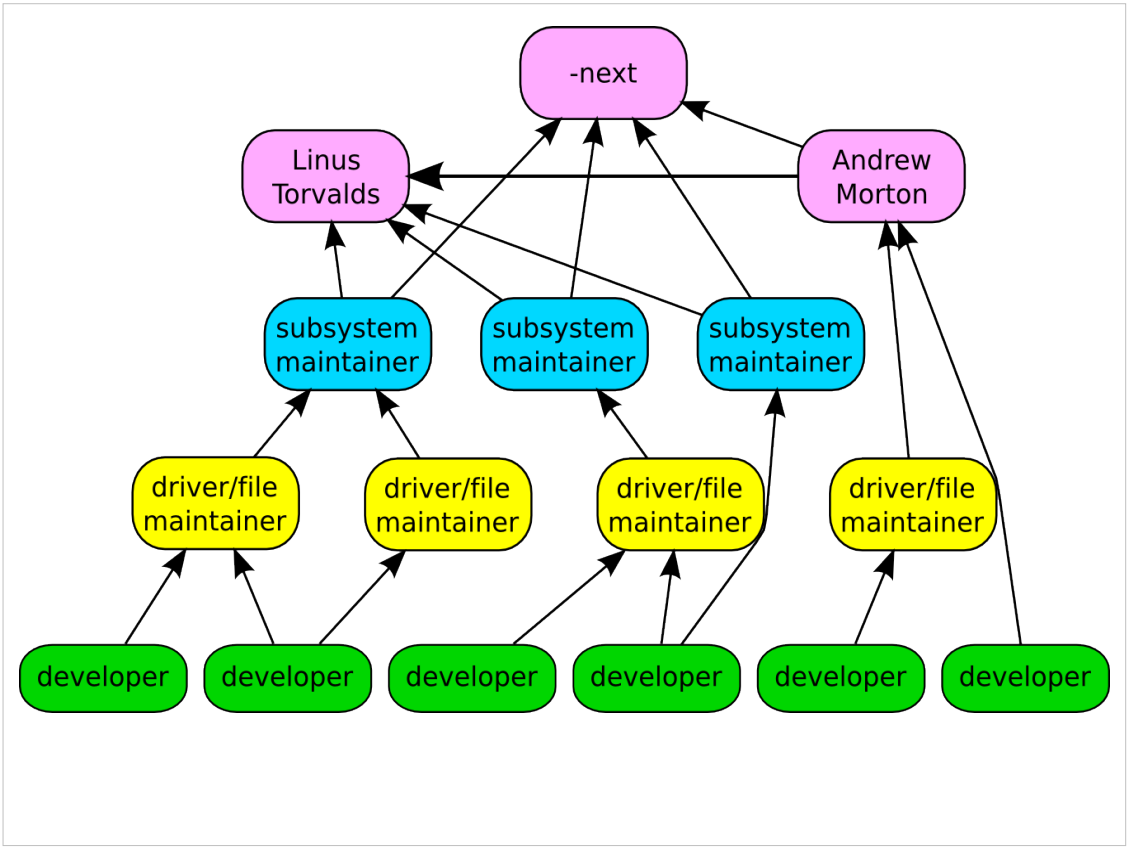
We have around 150 subsystem maintainers



Linux-next gets created every night from all of the different subsystem trees and build tested on a wide range of different platforms.

We have about 150 different trees in the linux-next release.

Andrew Morton picks up patches that cross subsystems, or are missed by others, and releases his -mm kernels every few weeks. This includes the linux-next release at that time.

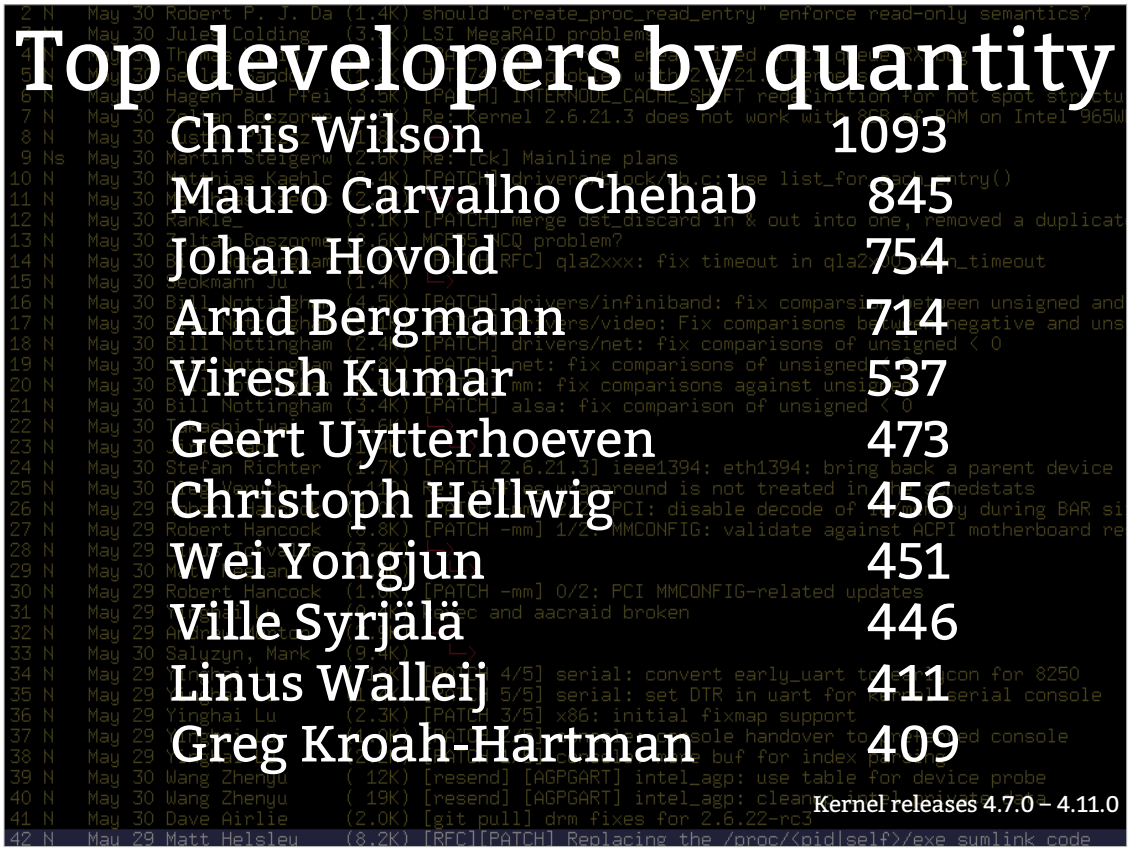


Every 3 months, when the merge window opens up, everything gets sent to Linus from the subsystem maintainers and Andrew Morton.

The merge window is 2 weeks long, and thousands of patches get merged in that short time.

All of the patches merged to Linus should have been in the linux-next release, but that isn't always the case for various reasons.

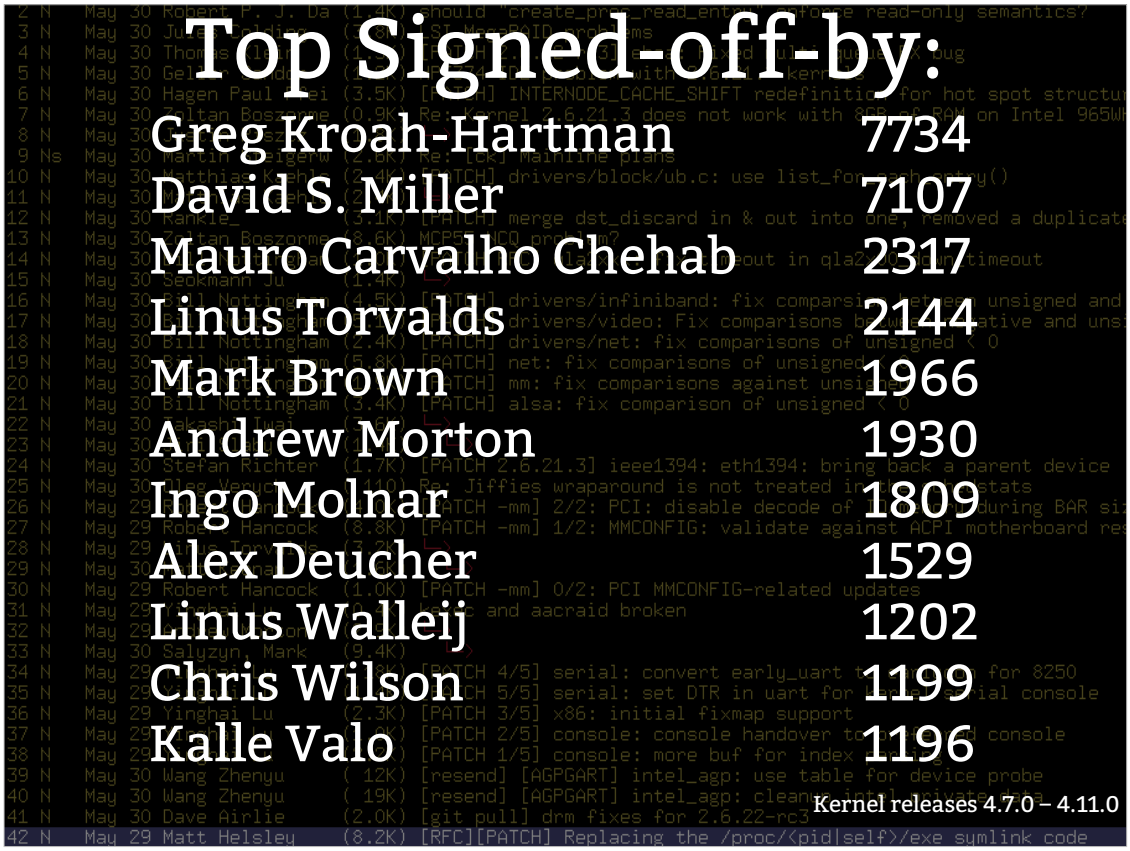
Linux-next can not just be sent to Linus as there are things in there that sometimes are not good enough to be merged just yet, it is up to the individual subsystem maintainer to decide what to merge.



Chris – intel graphics drivers
Mauro – Video 4 Linux (media drivers)
Johan – greybus, usb-serial, drivers
Arnd – janitorial cleanups and arch-generic
Viresh – greybus
Geert – janitorial
Christoph – vfs, filesystems, xfs, everywhere
Wei - Janitorial
Ville – intel graphics
Linus – gpio, pin, arm drivers
Greg – greybus

Top Signed-off-by:

Greg Kroah-Hartman	7734
David S. Miller	7107
Mauro Carvalho Chehab	2317
Linus Torvalds	2144
Mark Brown	1966
Andrew Morton	1930
Ingo Molnar	1809
Alex Deucher	1529
Linus Walleij	1202
Chris Wilson	1199
Kalle Valo	1196



Greg – driver core, usb, staging, greybus
David – networking, isa
Mauro – video 4 linux (media)
Linus – everything
Mark – embedded sound
Andrew – everything
Ingo – x86
Alex – radeon graphics
Linus – gpio and pinctl
Chris – intel graphics
Kalle – wireless drivers

Who is funding this work?

1. “Amateurs”	14.4%
2. Intel	13.4%
3. Red Hat	7.3%
4. Linaro	6.4%
5. IBM	3.4%
6. Samsung	3.4%
7. Consultants	3.0%
8. SuSE	2.9%
9. Google	2.7%
10. AMD	2.3%

Kernel releases 4.7 – 4.11

So you can view this as either 14% is done by non-affiliated people, or 86% is done by companies.

Now to be fair, if you show any skill in kernel development you are instantly hired.

Why this all matters: If your company relies on Linux, and it depends on the future of Linux supporting your needs, then you either trust these other companies are developing Linux in ways that will benefit you, or you need to get involved to make sure Linux works properly for your workloads and needs.

Who is funding this work?

11. Mellanox	1.9%
12. Renesas Electronics	1.8%
13. Huawei Technologies	1.6%
14. Oracle	1.6%
15. Broadcom	1.5%
16. Texas Instruments	1.5%
17. ARM	1.4%
18. Free Electrons	1.1%
19. Imagination Technologies	1.0%
20. NXP Semiconductors	0.9%

Kernel releases 4.7 – 4.11

Intel – 9000 patches
Huawei - 1115 patches (almost half done by one developer!!!)
NXP - 636

“Working upstream saves time and money”

Dan Frye – VP Open Systems, IBM
Dirk Hohndel – Chief Technologist, Intel

Getting involved

Run the kernel.org release on your machine

Getting involved

This book tells you how to build and install a kernel on your machine.

Free online

Getting involved

`Documentation/HOWTO`

`Documentation/development-process`

These documents in the kernel source directory are the best place to start if you want to understand how the development process works, and how to get involved.

The HOWTO file has links to almost everything else you ever wanted..

Getting involved

kernelnewbies.org

http://www.kernelnewbies.org

Getting involved

Google “write your first kernel patch”

This is a video of a talk I gave at FOSDEM, going through the steps, showing exactly how to create, build, and send a kernel patch.

Getting involved

kernelnewbies.org/KernelJanitors/Todo

So you know how to create a patch, but what should you do? The kernel janitors has a great list of tasks to start with in cleaning up the kernel and making easy patches to be accepted.

Getting involved

Linux Driver Project

`drivers/staging/*/TODO`

The staging tree also needs a lot of help, here are lists of things to do in the kernel for the drivers to be able to move out of the staging area.

Please always work off of the linux-next tree if you want to do these tasks, as sometimes they are already done by others by the time you see them in Linus's tree.

Getting involved

Eudyptula Challenge
(little penguin)

http://eudyptula-challenge.org/

Google “Linux kernel challenge” to find the site, if you can't remember Eudyptula.

It is a series of programming challenges, all run through email that starts out with a “Hello World” kernel module, and gets more complex from there. Over 4000 people are currently taking the challenge, and is a lot of fun if you don't know where to start out.

You need knowledge of C, but that's about it.

Obligatory Penguin Picture