



Upstreaming

Lee Jones - Linaro
Team Lead, ST-Ericsson Landing Team
May 2011



Who turned up?

- Maintainers & current contributors
- Would be upstreamers
- Don't know what upstreaming is
- People against upstreaming



Topics

- Repository hierarchy
- Maintainers
- Benefits of upstreaming
- How to upstream your code
- How long does upstreaming take?



Repository hierarchy

- Theoretical hierarchy of repositories
- Git maintained
- Each repository specialises in one topic (902)
- One maintainer per topic (505)



Repository hierarchy

Mainline

Architectures

arm

cris

mips

sh

x86

... (24)

Driver Sub-Systems

ata

base

dma

i2c

usb

... (90)

Sub-arches

mx6

ndk

sam

omap

?

Could be you

...

Dev

More specific driver subsystems

promise

power

ioat

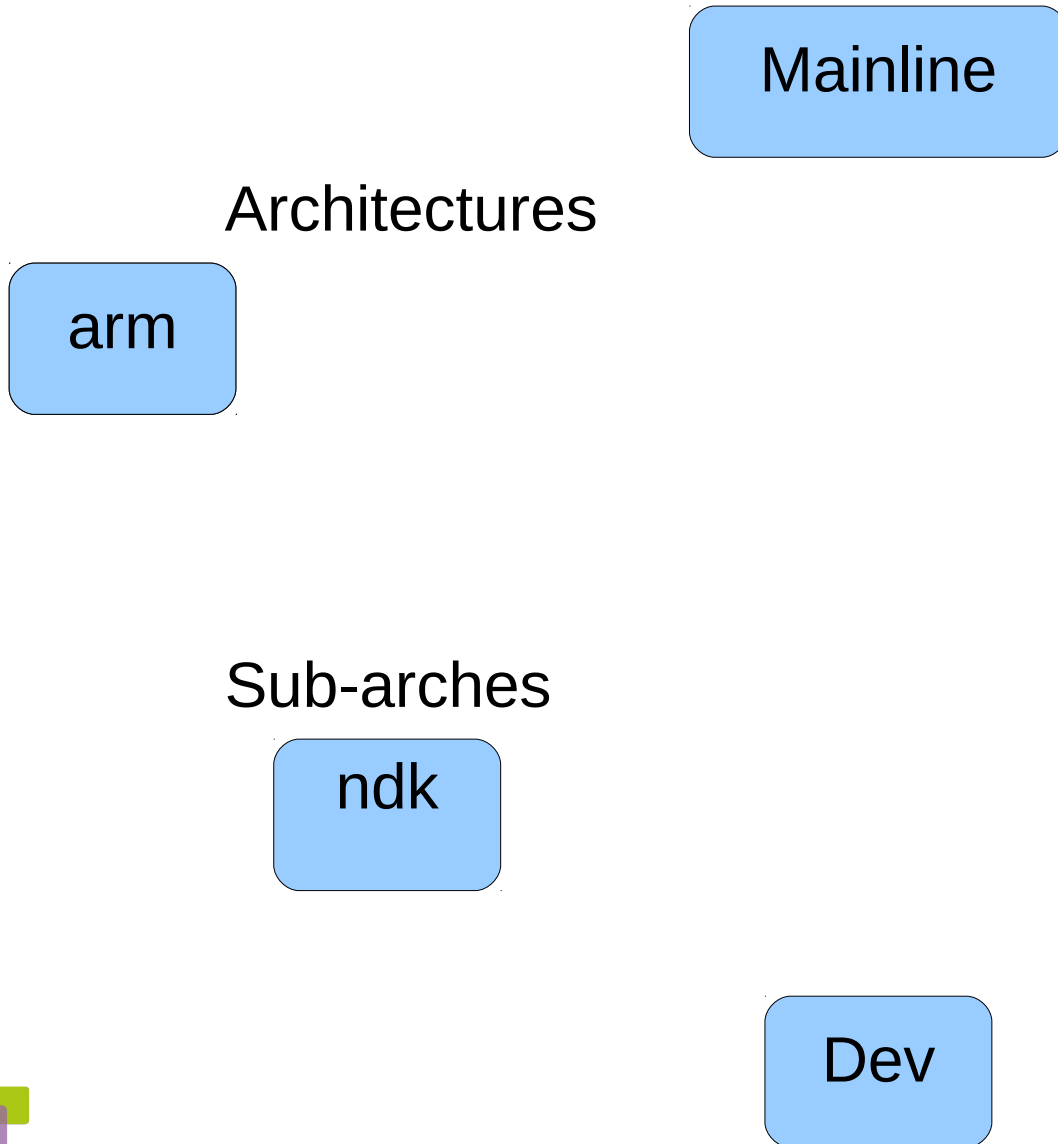
scx200

ohci

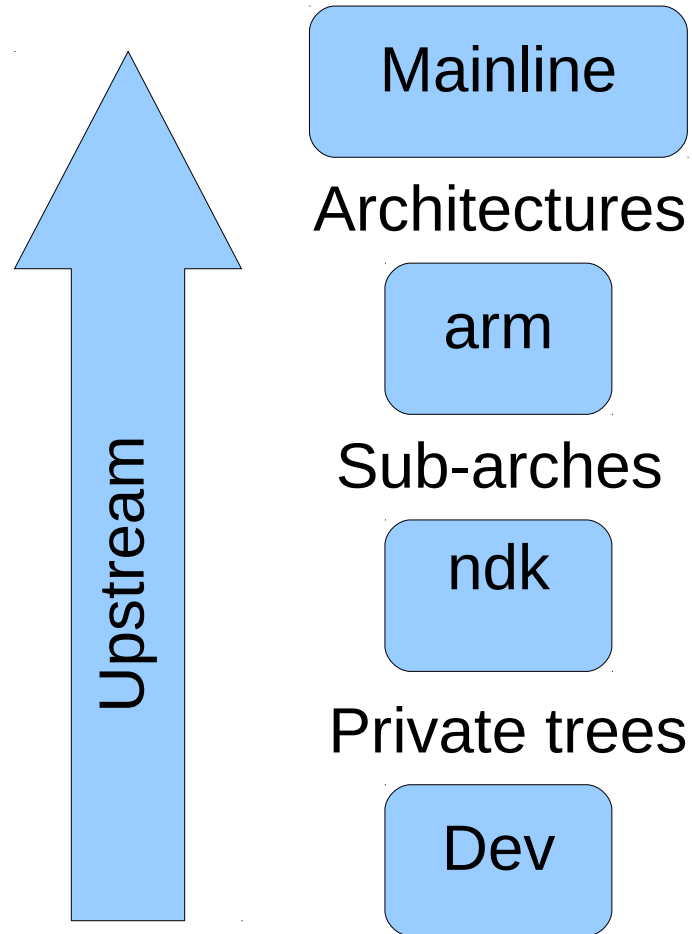
...



Repository tree hierarchy



Repository tree hierarchy



Topics

- Repository hierarchy
- Maintainers
- Benefits of upstreaming
- How to upstream your code
- How long does upstreaming take?

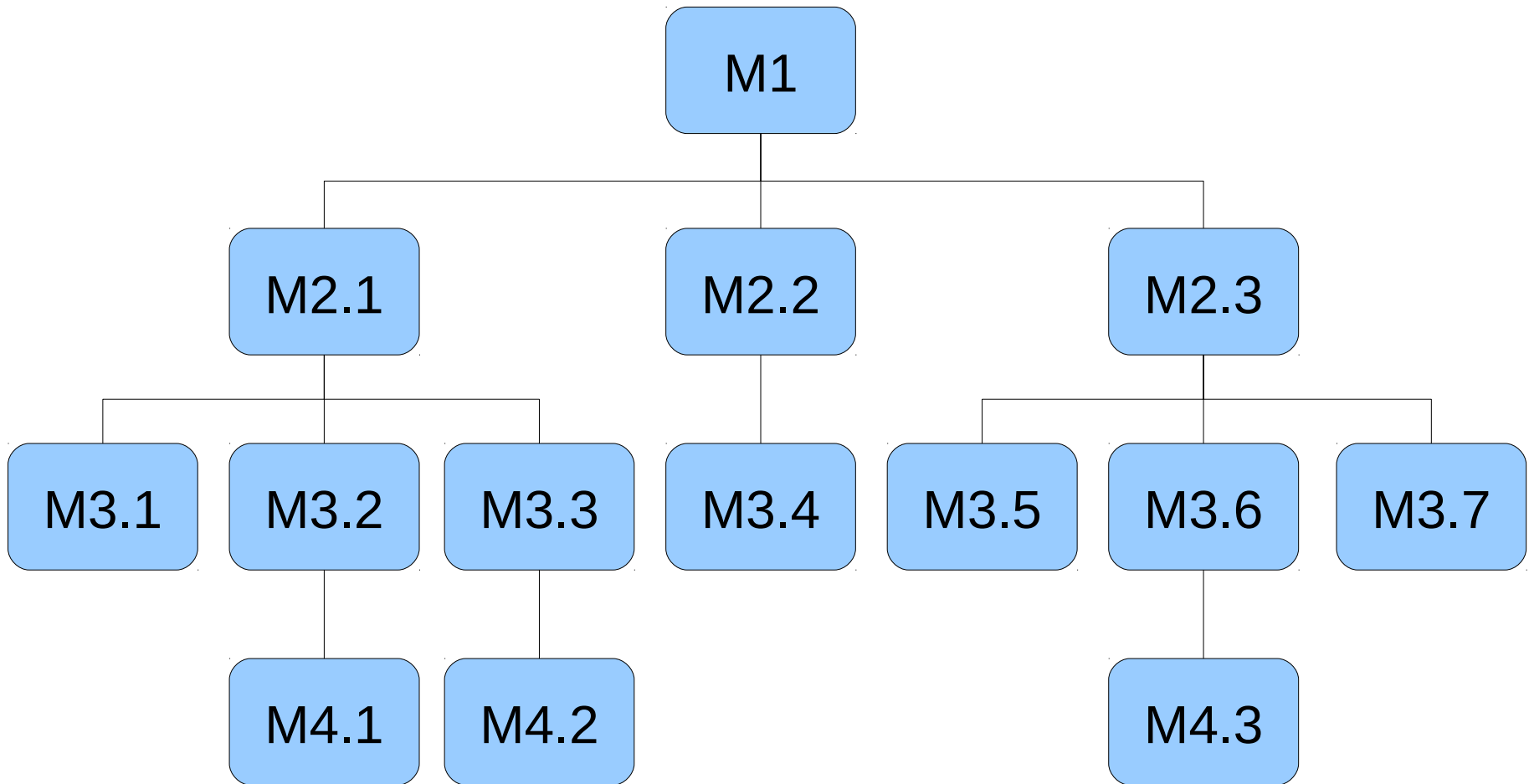


Maintainers

- Subsystem code owners
- Gatekeepers
- Upstreaming responsibilities



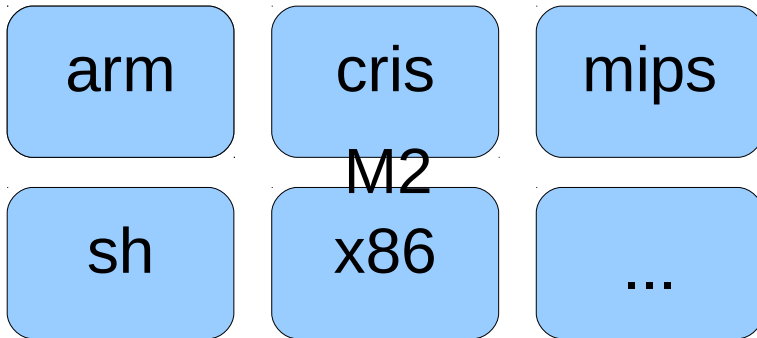
Maintainer hierarchy



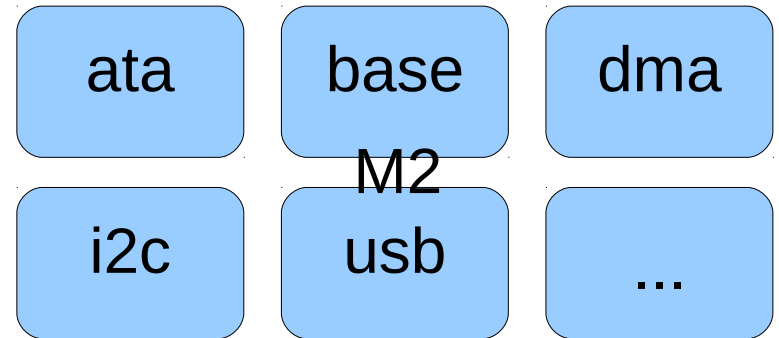
Repository tree hierarchy

Mainline
M1

Architectures



Driver Sub-Systems



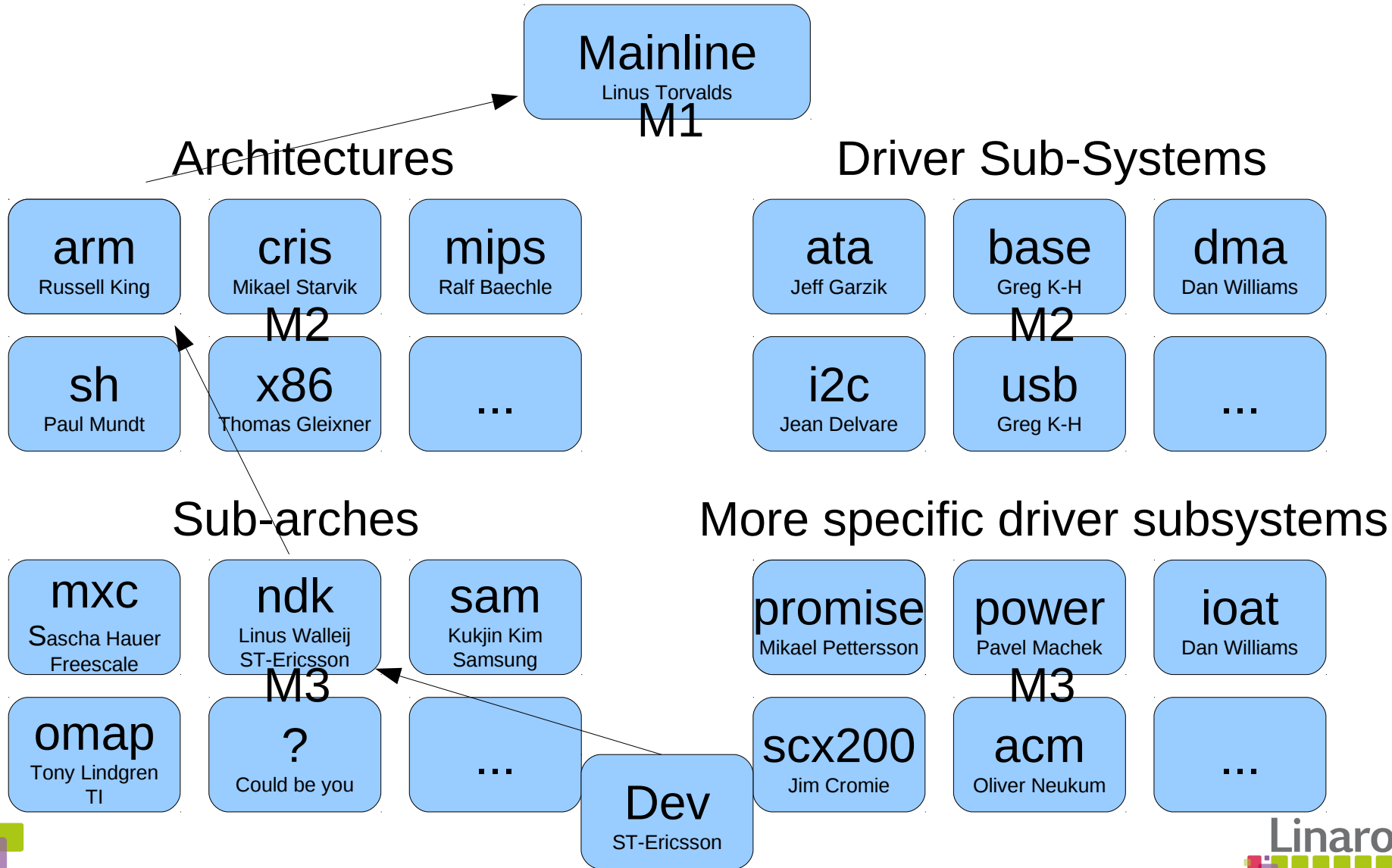
Sub-arches



More specific driver subsystems



Maintainer hierarchy



Maintainers

- Stats

- 505 unique maintainers
- 902 subsystems/topics

- LINUX/MAINTAINERS

ARM/NOMADIK ARCHITECTURE

M: Linus Walleij <linus.walleij@stericsson.com>

M: Alessandro Rubini <rubini@unipv.it>

M: STEricsson <STEricsson_nomadik_linux@list.st.com>

L: linux-arm-kernel@lists.infradead.org (moderated for non-subscribers)

S: Maintained

F: arch/arm/mach-nomadik/

F: arch/arm/plat-nomadik/

F: drivers/i2c/busses/i2c-nomadik.c

T: git git://git.kernel.org/pub/scm/linux/kernel/git/linusw/linux-stericsson.git



Topics

- Repository hierarchy
- Maintainers
- Benefits of upstreaming
- How to upstream your code
- How long does upstreaming take?



Benefits of upstreaming

- Current attitude to upstreaming
 - Waste of resources
 - Time
 - Engineering power
 - Giving away work to competitors

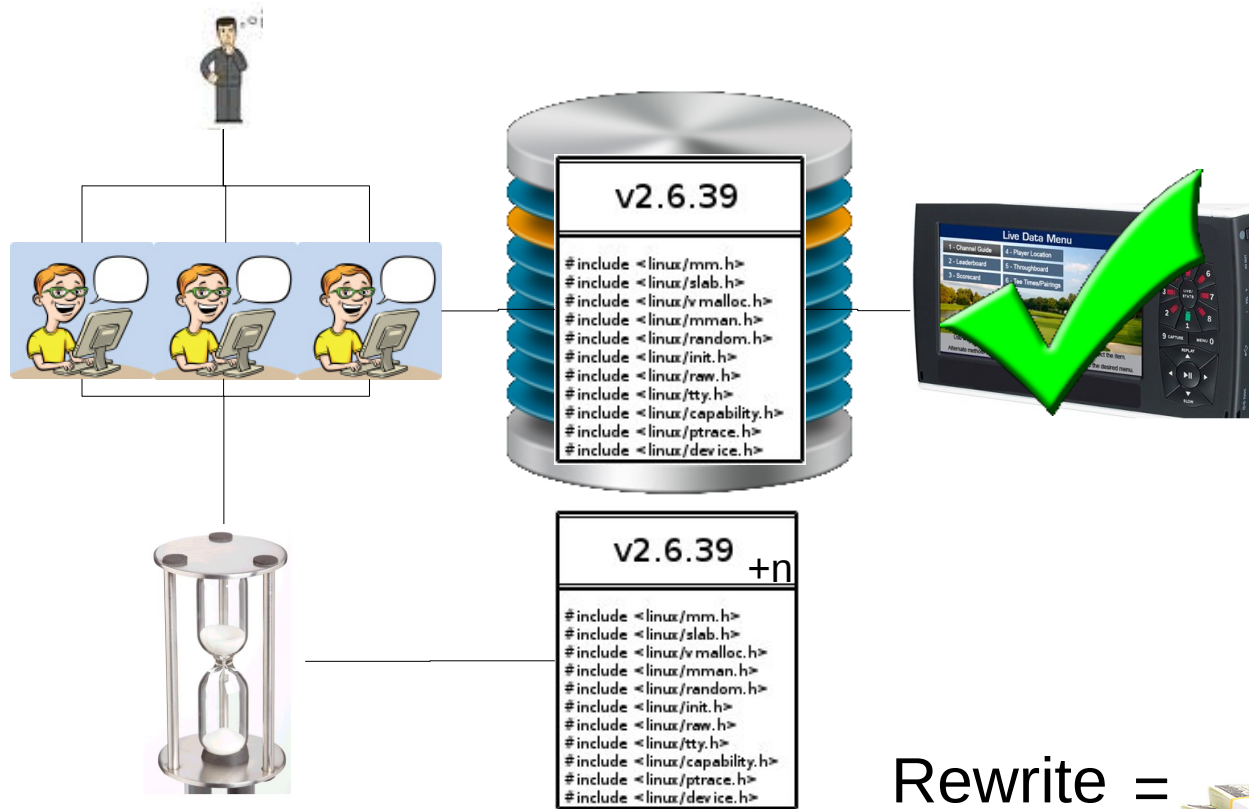


Benefits of upstreaming

- Maintainability
 - Responsibility
 - Forward porting
 - Future proof



Future proof



Benefits to upstreaming

- Maintainability
- Quality Assurance



Benefits to upstreaming

- Quality Assurance
 - Compulsory reviews
 - Programmed correctly
 - Optimised
 - Less bugs
 - Testing and validation on a massive scale
 - Direct user feedback
 - Bug reports
 - Contact email in file header
 - ``git log`` or ``git blame``



Topics

- Repository hierarchy
- Maintainers
- Benefits of upstreaming
- How to upstream your code
- How long does it take?



How to upstream your code

- Preparation

- Plan

- Discuss unknowns with the MLs

- Author

- LINUX/Documentation/CodingStyle



How to upstream your code

- Create patches
 - ``git format-patch`` or ``diff -purN``
 - Logical groups of functionality
 - Individually compilable
 - Keep small & upstream often
 - Easier to understand & review
 - Easier to make amendments and resubmit
 - Identifies fundamental issues early



How to upstream your code

- Preparation
- Review



How to upstream your code

- Review
 - Ensure they compile on the latest kernel
 - Test functionality
 - Run `LINUX/scripts/checkpatch.pl`
 - Send to internal mailing lists
 - Once complete add Signed-off-by



How to upstream your code

- Preparation
- Review
- Submission



How to upstream your code

- Submission
 - RTFM and take heed
 - LINUX/Documentation/SubmitChecklist
 - LINUX/Documentation/SubmittingPatches
- Locate e-mail destination
 - Browse LINUX/MAINTAINERS
 - Run LINUX/scripts/get_maintainer.pl
 - Maintainer's e-mail address
 - Mailing list address
 - Affected or interested parties
 - CC LKML linux-kernel@vger.kernel.org



How to upstream your code

- Submission
 - Sending patches
 - ``git send-email``
 - Email client
 - [LINUX/Documentation/email-clients.txt](#)



How to upstream your code

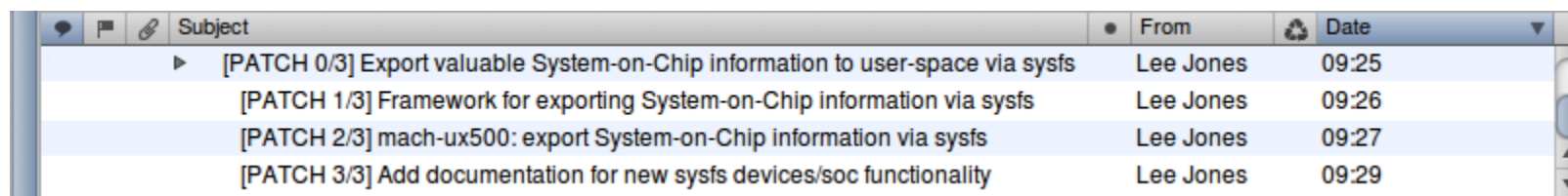
- Submission
 - Single patch



A screenshot of an email client interface. At the top, there is a 'Quick Filter' bar with icons for search, flags, attachments, and a search input field containing 'Filter these messages... <Ctrl+F>'. Below this is a table with columns for 'Subject', 'From', and 'Date'. A single email entry is visible.

Subject	From	Date
[PATCH] ux500: Add Rohm BH1780GLI Light Sensor to i2c_board_info	Lee Jones	11/01/11 13:10

- Patch set



A screenshot of an email client interface showing a series of related patches. The table has columns for 'Subject', 'From', and 'Date'. The first entry is expanded, showing a sequence of four patches.

Subject	From	Date
▶ [PATCH 0/3] Export valuable System-on-Chip information to user-space via sysfs	Lee Jones	09:25
[PATCH 1/3] Framework for exporting System-on-Chip information via sysfs	Lee Jones	09:26
[PATCH 2/3] mach-ux500: export System-on-Chip information via sysfs	Lee Jones	09:27
[PATCH 3/3] Add documentation for new sysfs devices/soc functionality	Lee Jones	09:29



Dealing with the mailing lists

- Be patient
 - Your patch may not be reviewed immediately
- Be polite and don't take offense
 - Some maintainers are hard to work with
 - Straight talking and non-diplomatic
- Explain the reasons for your decisions
 - Maintainers aren't always right
- If you don't understand, ask
 - "He who asks a question is a fool for a minute; he who does not, remains a fool forever." --

Chinese proverb



Dealing with the mailing lists

- Getting flamed

- “Christoph said my code is buggy, my office smells and my hair looks strange. He said it on linux-kernel and everybody saw it..”,

kernelnewbies

- Congratulations!



How to upstream your code

- Preparation
- Review
- Submission
- Make changes & re-submit



How to upstream your code

- Make suggested changes & re-submit
 - Unlikely to be correct first attempt
 - Don't be discouraged
 - Typical reasons for change requests
 - Use or overuse of `#ifdefs`
 - Coding style
 - Use of `MACROs` instead of `'static inline'`
 - Use of `char*` instead of `const`
 - Incorrect use of APIs or hacky code



Topics

- Repository hierarchy
- Maintainers
- Benefits of upstreaming
- How to upstream you code
- How long does upstreaming take?



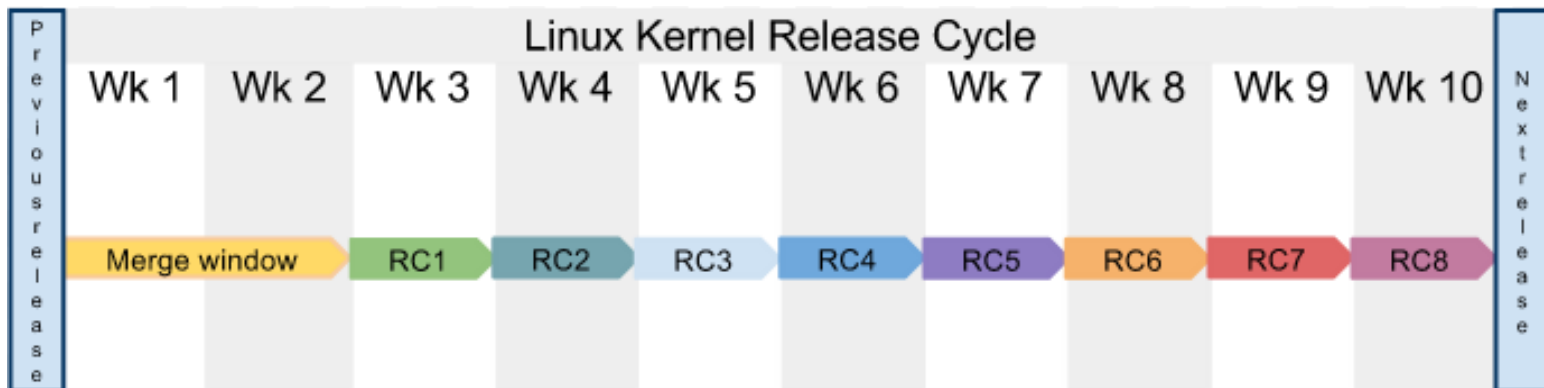
How long does upstreaming take?

- Factors
 - Code corrections
 - How much needs to be changed
 - Change complexity
 - Merge window

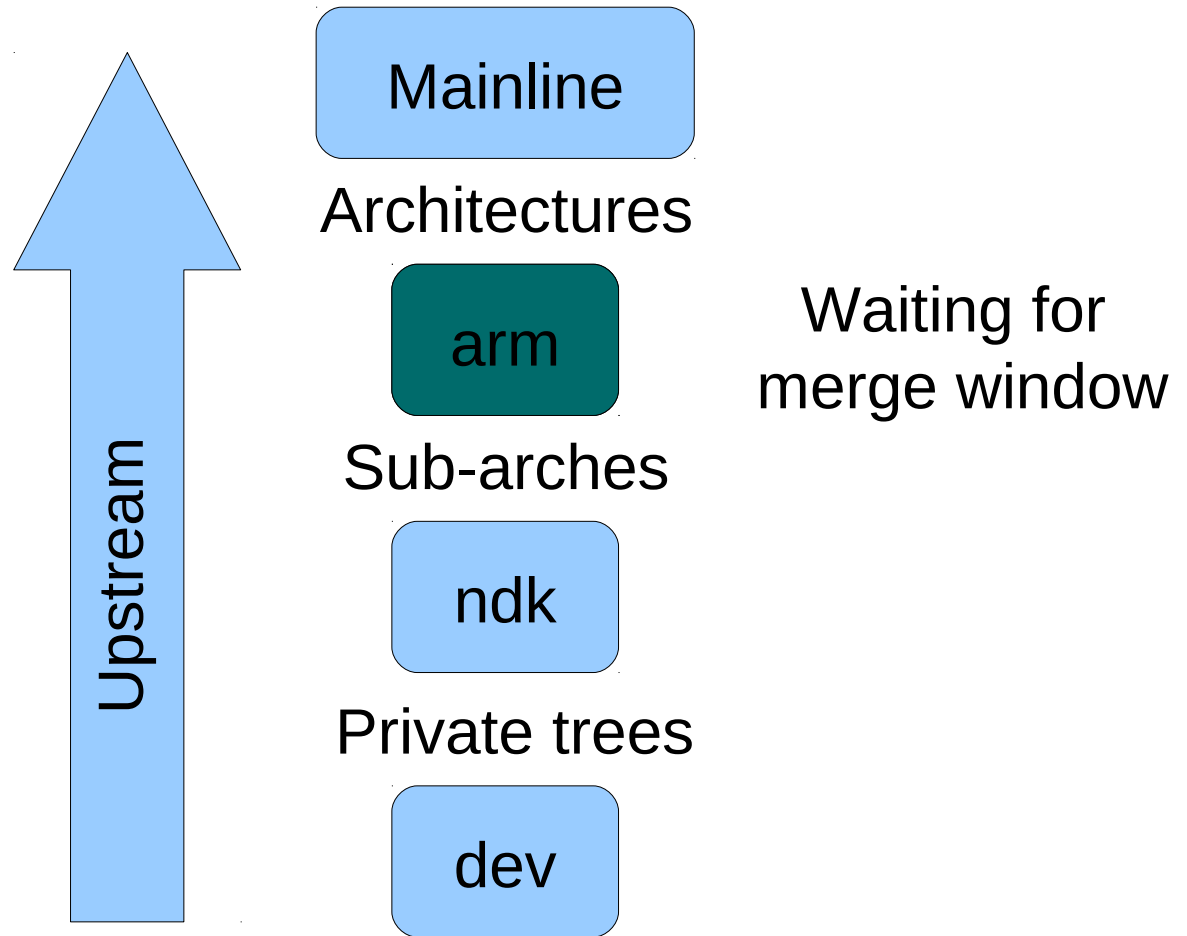


How long does upstreaming take

- Merge window
 - Usually every 8-10 weeks
 - Open for 2 weeks



Merge window

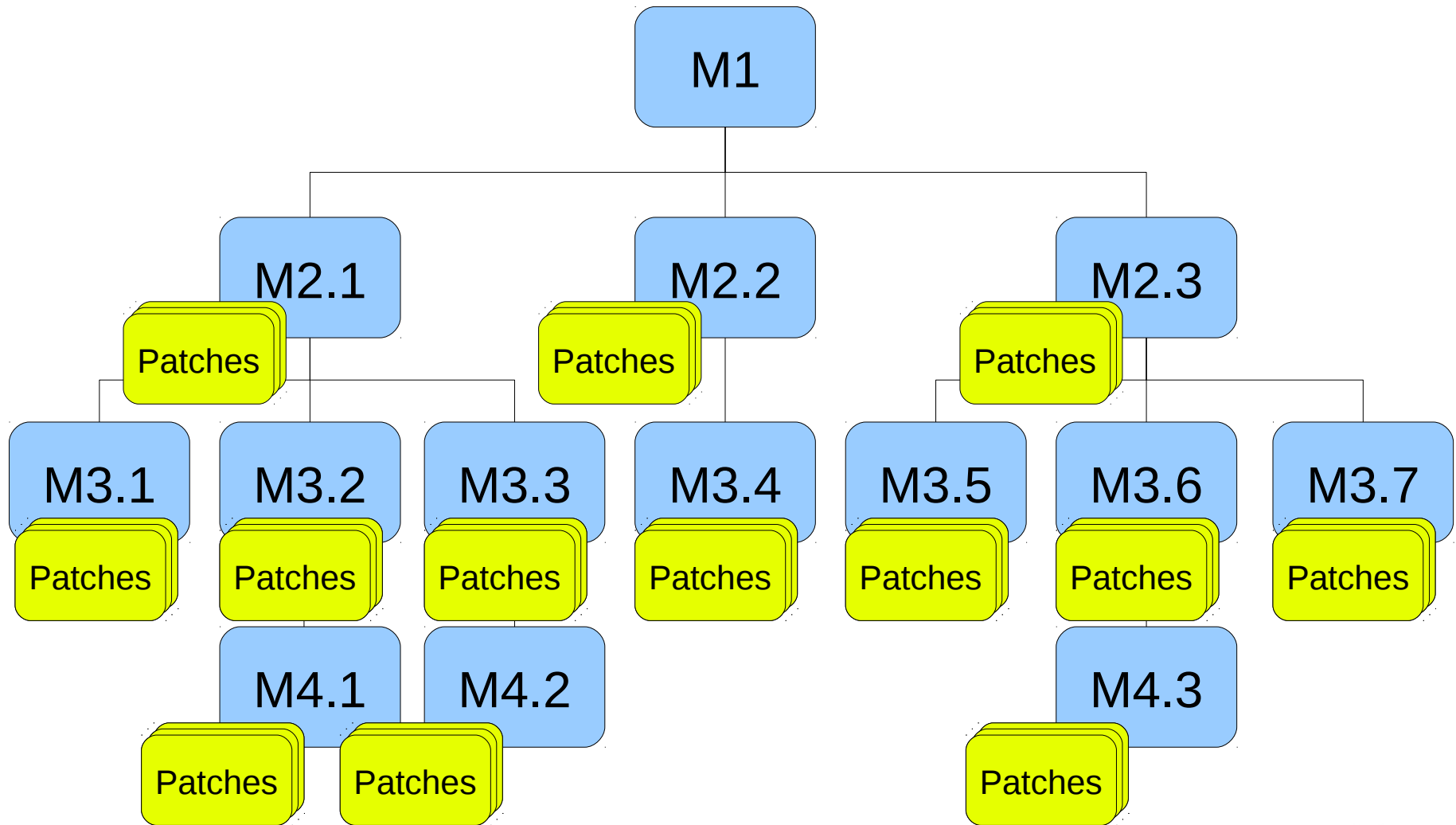


How long does upstreaming take?

- Factors
 - Code corrections
 - How much needs to be changed
 - Change complexity
 - Next merge window
 - Depth in maintainer hierarchy



How long does upstreaming take?



How long does upstreaming take?

- Factors
 - Code corrections
 - How much needs to be changed
 - Change complexity
 - Maintainer count
 - Next merge window
 - How you're perceived on the MLs



To conclude

- Non-propitiatory code is always worth upstreaming
- Upstream little and often
 - Constant, world class reviews
 - Avoid large re-writes
 - Enjoy the process
- Act professional and knowledgeable on the MLs – make people **want** to help you
- Enjoy the process



Questions?

