

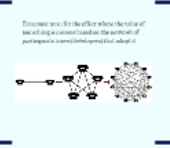
Open Source

How it works

Developers publish their software online.
System only works when people fix what they publish.
They need to cooperate.



Network Effects



Types



Explanatory Power

- Format wars
- Subsidies to one side of a market
- TV - relationship of advertisers and viewers
- Fanboy behavior

Network Effects are Everywhere

- All large categories of consumer and enterprise leverage network effects
- Google, Facebook, etc.
- properties increase your value
- your power advantage
- Google+ opened videos to view a large view
- Applies to anyone who publishes a platform, where other developers or users create value.

Embedded

Dedicated function

- Router
- TV
- Digital camera
- Set-top box
- Robots

Tension between generalization and specialization

History

- Evolution to general purpose OS
- 1960s - 1970s
- 1980s - 1990s
- 2000s - 2010s
- 2010s - 2020s

General Purpose Hardware

- Transition to developing software hardware resources
- Software is the new hardware
- Hardware is the new software
- Software is the new hardware
- Hardware is the new software

Internet of Things

IOT Changes the equation

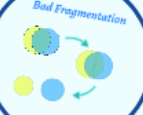
- We wear computers in our:
 - cars
 - appliances
 - furniture
 - light switches
 - clothing
- Possibly in our bodies and our food!
- In our infrastructure - monitoring environment, water, energy, traffic
- Want to run Linux on a 10-cent processor, that runs for years on a single charge

Linux in IOT



Fragmentation

Bad Fragmentation



Good Fragmentation



Linus Quote



© 2010-2011 Linus Torvalds
This document is licensed under the Creative Commons Attribution-ShareAlike license
http://creativecommons.org/licenses/by-sa/4.0/

The Paradox of Embedded and Open Source

Tim Bird

Senior Software Engineer, Sony Mobile Communications, Inc
Chair, CE Workgroup Architecture Group
(Founder of ELC)

Keynote

- I haven't keynoted this event for several years
- I don't really have an action for you...



A FILM BY CHRISTOPHER NOLAN

INCEPTION

4/29/2014

FROM THE DIRECTOR OF THE DARK KNIGHT

The Paradox of Open Source and Embedded

ELCeption

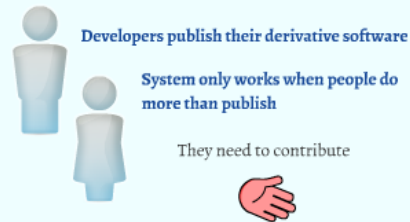
- In the movie Inception, an international team implants ideas into their subjects minds, by invading their dreams.
- I want to inject an idea into your mind
 - But, I don't care if you know I'm trying to do it
 - And I don't have my dream-invading equipment with me...

Outline

- Open Source and network effects
- Embedded
- Internet of Things
- Fragmentation

Open Source

How it works



Definition
Open Source Software can be freely used, changed and shared by anyone

Legal framework
GPL License
Guarantees freedom for users and developers to modify the source code

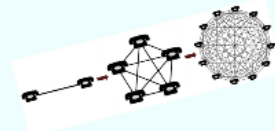
Other Licenses
• BSD
• Apache
• MIT

Different Legal Terms

Creates a community



Network Effects



Open Source and Network Effects

Other developers write software you use

More developers = more value

Ecosystem

Related services: Books, training, tools, jobs

Community size

• Not a single community
• Lots of sub-communities

Definition

Open Source Software can be freely used, changed and shared by anyone

GPL License

guarantees freedom
for downstream users
to view source code



framework

Other Licenses

GPL License



Guarantees freedom
for downstream users
to obtain source code

Legal framework

Other Licenses

- BSD
- Apache
- MIT

Different Legal Terms



Open Source

How it works



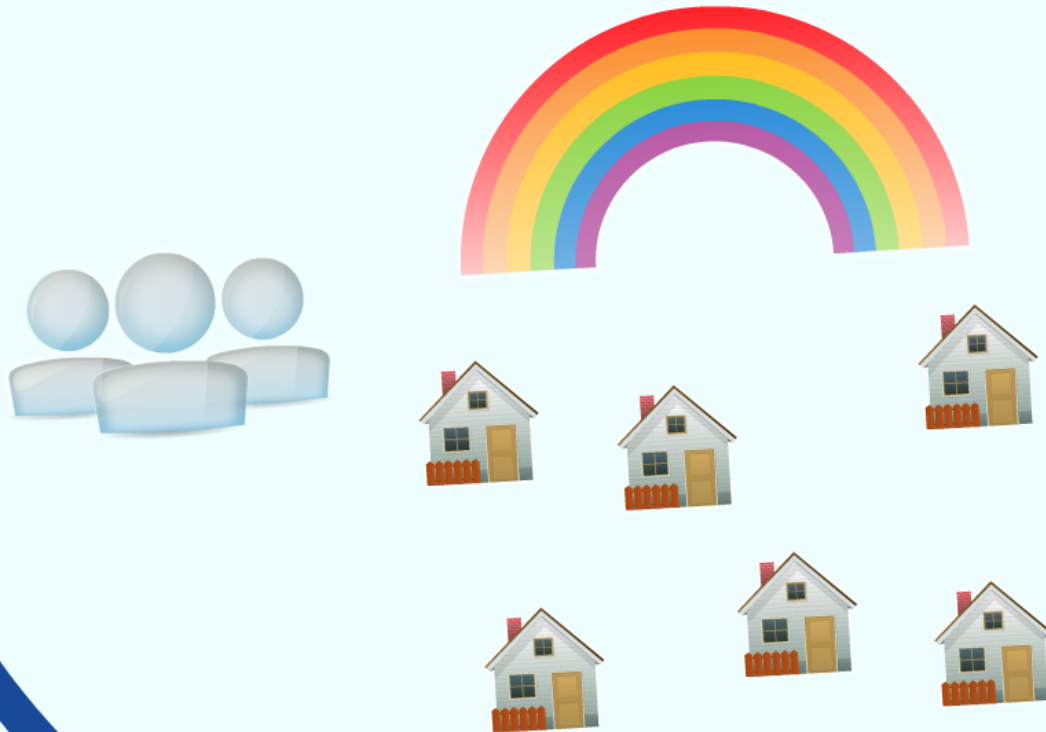
Developers publish their derivative software

System only works when people do more than publish

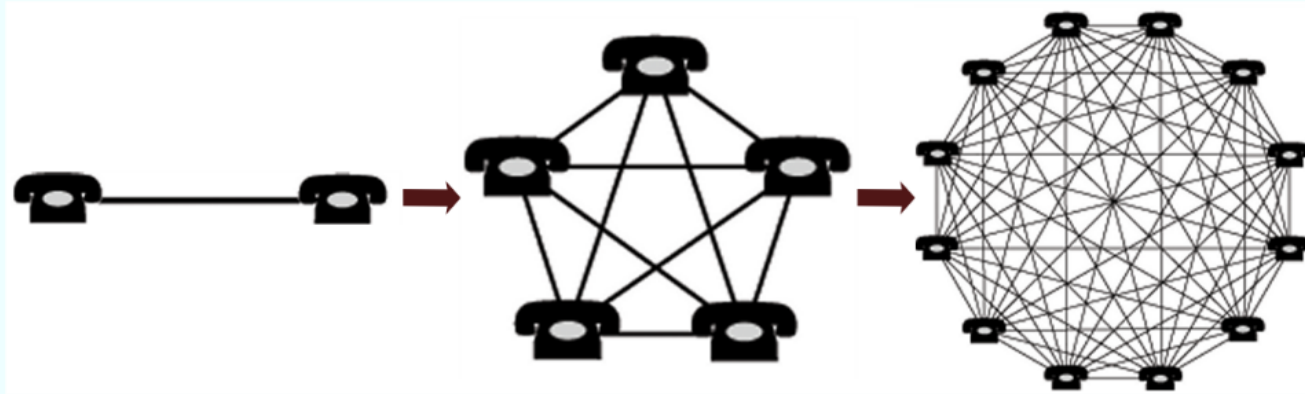
They need to contribute



Creates a community



Network Effects



Network Effects

Economic term for the effect where the value of something increases based on the network of participants (users/developers) that adopt it.



Explanatory Power

- Format wars
- Subsidies to one side of a market



TV - relationship of advertisers and viewers

- Fanboy behavior

Convincing people to use your platform is a rational behavior -- even if irrational arguments are used



Network Effects are Everywhere

All large companies understand and try to leverage network effects

Google



3rd parties increase your value!

First mover advantage

Companies spend billions to win a format war



Applies to anyone who publishes a platform, where other developers or users create value.

Types



Format Wars



Operating Systems



Open Source and Network Effects

Other developers write software you use

More developers = more value



Related services: Books, music, tools, etc.

Community size

- Not a single community
- Lots of sub-communities

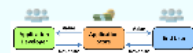
Size matters

Some communities are very small



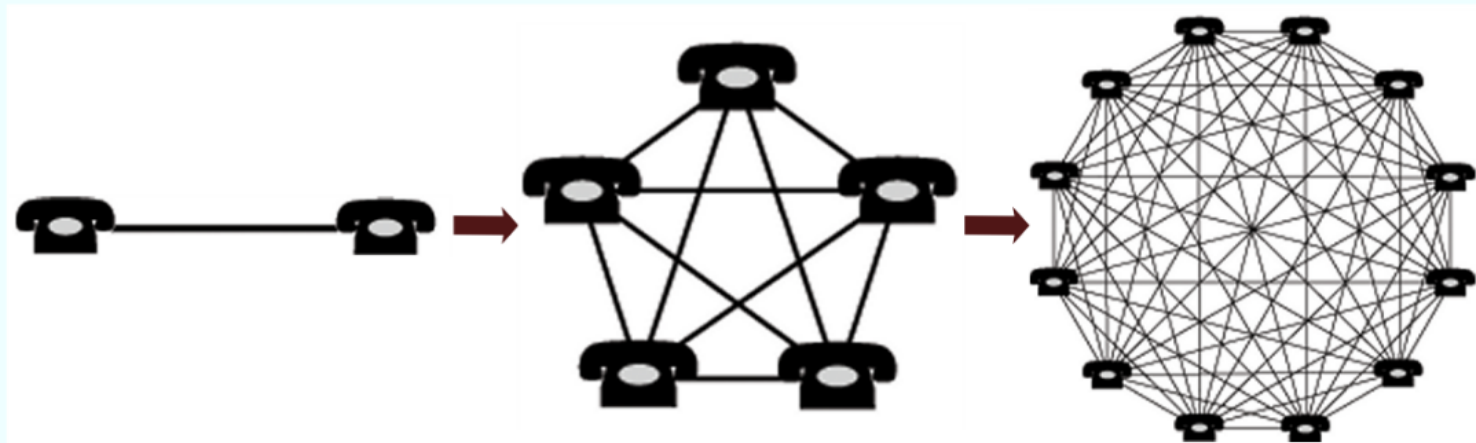
People are in multiple communities simultaneously

Two-sided markets



In this -- all existing and future users create value

Economic term for the effect where the value of something increases based on the network of participants (users/developers) that adopt it.



Types

Phone network



Format Wars

- VHS vs. Betamax
- HD DVD vs Blu-ray



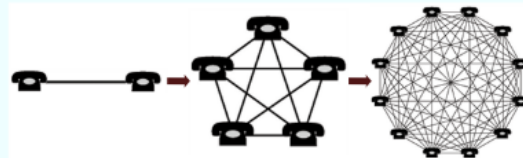
Operating Systems

- Windows vs. Mac vs. Linux
- Android vs. iOS



iOS

Phone network



Operating Systems

- Windows vs. Mac vs. Linux
- Android vs. IOS



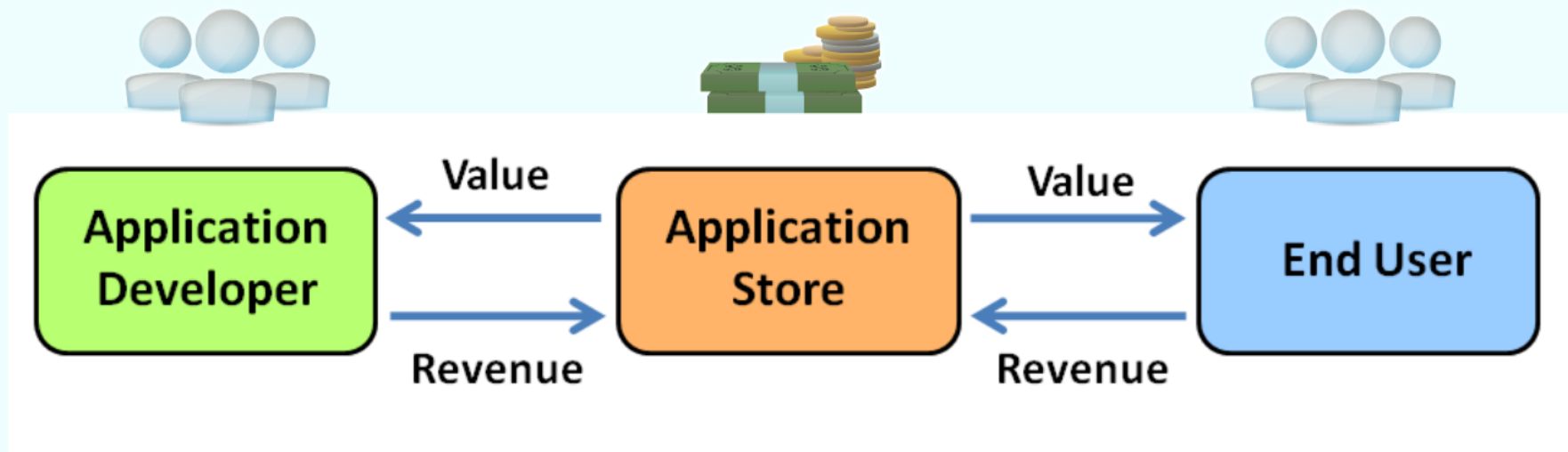
iOS

Format Wars

- VHS vs. Betamax
- HD DVD vs Blu-ray



Two-sided markets



In 2014 -- still evolving our theories
about multi-sided markets

Network Effects are Everywhere

All large companies understand and try to leverage network effects

The Google logo, featuring the word "Google" in its characteristic multi-colored font.The Microsoft logo, featuring the word "Microsoft" in a bold, black, sans-serif font.

3rd parties increase your value!

First mover advantage

leverage network effects

Google



Microsoft

3rd parties increase your value!

First mover advantage

Companies spend billions
to win a format war



3rd parties increase your value!

First mover advantage

Companies spend billions
to win a format war



Applies to anyone who publishes a
platform, where other developers or
users create value.

Explanatory Power

- Format wars
- Subsidies to one side of a market



TV - relationship of
advertisers and viewers

- Fanboy behavior

Convincing people to use
your platform is a rational
behavior -- even if irrational
arguments are used



where

Open Source and Network Effects

Other developers write software you use

More developers = more value

Ecosystem



Related services:
Books, training,
tools, jobs



Community size

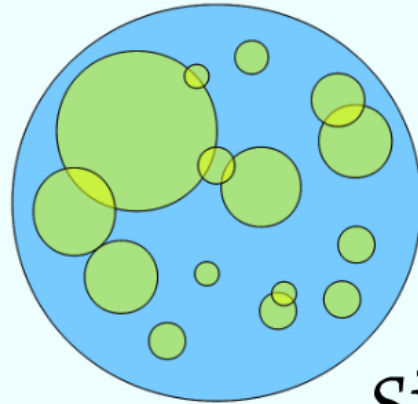
of a single community

Ecosystem



Related services:
Books, training,
tools, jobs

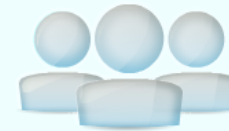
Community size



- Not a single community
- Lots of sub-communities

Size matters

Some communities
are very small



People are in multiple
communities simultaneously

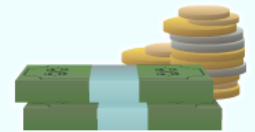


Cost to contribute

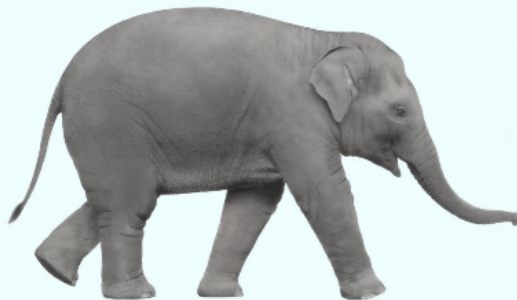
To build community, must create generalized software



Generalized software costs more to write



Doesn't perform as well



Bigger
Slower



Embedded

Dedicated function

- Router
- TV
- Digital camera
- Set-top box
- Robots

Mobile phones



*Tension between generalization
and specialization*



VS



Re-Specialization



Andrew Murray's talk on Bo
(2010 ELC Europe)



Amulet

History

From custom to general-purpose OS

- VxWorks
 - Nucleus
 - pSOS
 - VRTX
 - LynxOS
 - ITRON
 - QNX
-
- Linux

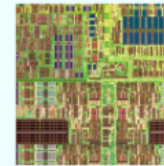


Cheaper, more ca
hardware

General Purpose Hardware

Tradeoff in development time vs. hardware resources

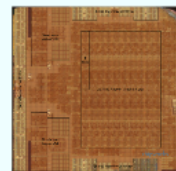
Modern SOC's have enormous complexity



Cheapest DRAM is 32M



CPU with 9 cores is same cost as one with 3 cores



Get used to wasting silicon!

, more capable
re

• Ex.
• Work. Exports
• 4 years of work.
• Good track record.
→ Right time to start.
→ I also want to share my observations.
→ I also want to share my observations.

Re-Specialization



Andrew Murray's talk on Boot Time
(2010 ELC Europe)

The Right Approach to Minimal Boot Times

Andrew Murray
Senior Software Engineer, NPC Data

- Driver and kernel development
- Embedded applications development
- Windows driver development

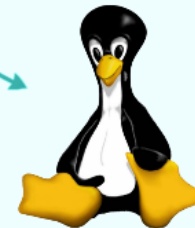
Work Experience

- 4 years of experience working with embedded U
- Good track record in dramatically reducing costs through NPC Data's best time reduction service
- Tight timescales often doesn't permit nice, fit solutions - however this frustration had provided ideas I wish to share today
- I also wish to share my observations and my time reduction

The swiftBOOT Approach

- Further improvements can be gained in different times:
- Parallelisation
- Using Argon's async framework
- Deferred loading of less important
- Loadable kernel modules

Linux Kernel (Before and After)



vs. hardware resources

those Hardware

Device Tree



Not a rant
OK - a little bit of a rant

DT Problems

- No typing
- Separation of data from code
 - Harder to write drivers
- Language problems
- Unfamiliarity
- Multi-node dependencies not easy-to-express

DT is Hard to specialize

It's meant to support single image

Kernel parses tree at runtime

Can't do compile-time optimizations

Relate here my long sad story about Link-Time Optimisation and how parsed data items are not optimizable by the compiler.

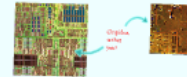
DT and network effects

DT helps build network effects



Has encouraged restructuring platform code for re-use

Exposes IP blocks between platforms



all code
ivers

DT is Hard to specialize

It's meant to support
single image

Kernel parses tree
at runtime

Can't do compile-
time optimizations

*Relate here my long sad story about Link-Time
Optimization and how parsed data items are not
optimizable by the compiler.*

not

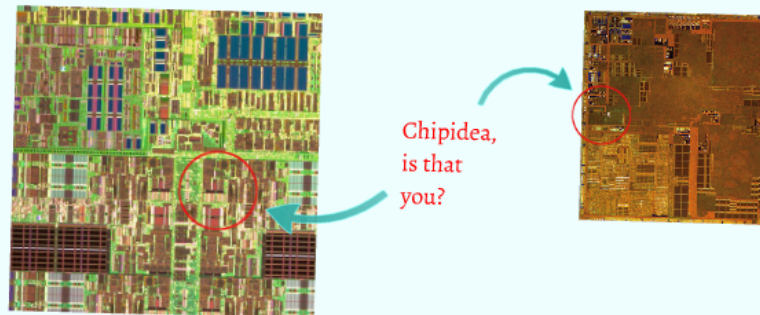
DT and network effects

DT helps build network effects

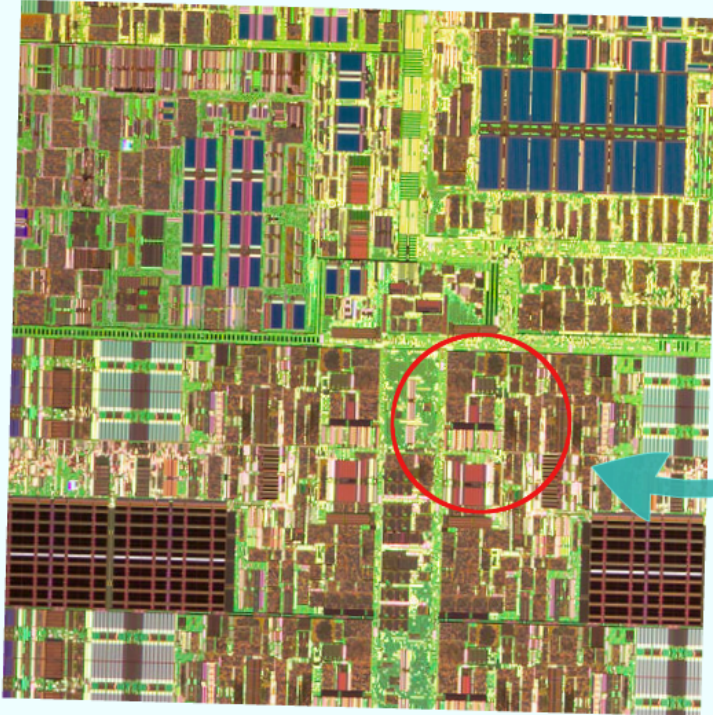


Has encouraged restructuring
platform code for re-use

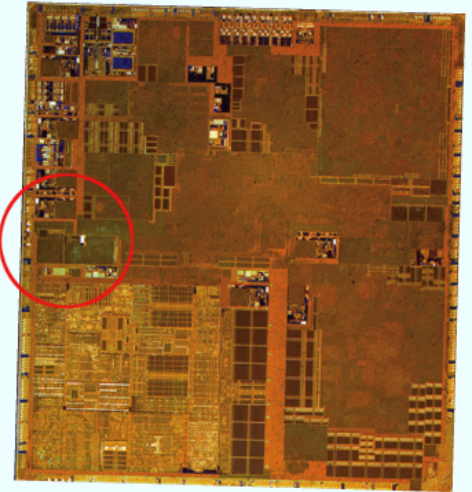
Exposes IP blocks between platforms



Exposes IP blocks between platforms



Chipidea,
is that
you?



Internet of Things

IOT Changes the equation

We want computers in our:

- cars
- appliances
- furniture
- light switches
- clothing



Possibly in our bodies
and our food!



In our infrastructure - monitoring
environment, water, energy, traffic



Want to run Linux on a 10-cent
processor, that runs for years
on a single charge

Linux in IOT

Why Linux?

Do we actually need Linux here?



Reuse

What reuse are we talking for?

- Embedded Linux
- Embedded Linux
- Embedded Linux
- Embedded Linux



Streamlining Linux

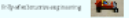
Modularity/Legos

Large blocks of software



Subtractive Engineering

Infrastructure engineering



Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Infrastructure engineering

Finally - Linux on a cereal box



Linus Quote



Linux Mail - Ask Linus, May 2000

IOT

Changes the equation

We want computers in our:

- cars
- appliances
- furniture
- light switches
- clothing



Possibly in our bodies
and our food!



In our infrastructure - monitoring
environment, water, energy, traffic



Want to run Linux on a 10-cent
processor, that runs for years
on a single charge

Pro

Linux is too
• Big
• Slow
• Power-hungry
• Insecure

Finally - Linux on a ceral box



Linux in IOT

Why Linux?

Do we actually need Linux here?



Reuse

What re-use are we striving for?

- People want to leverage:
- Network stack
 - File systems
 - USB
 - NFC

SOC support



Streamlining Linux

Modularity/Legos

Lego Model of Software



Subtractive Engineering

Folly of subtractive engineering



As system scales up, it's harder to remove than build from scratch

Nobody wants to remove stuff they don't understand



Just say "NO" to subtractive engineering

Problem with Linux in IOT

Linux is too:

- Big
- Slow
- Power-hungry
- Insecure

Linux 0.11 system
ran in 2MB

More features since then

Lose community

If we slim down Linux,
it's not Linux anymore

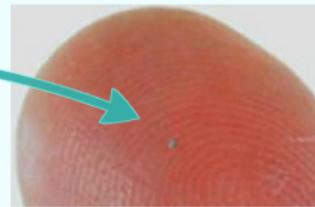
No network effects

in our bodies
Food!



Why Linux?

Do we actually need Linux here?



Hitachi rfid chip

Problem with Linux in IOT

Linux is too:

- Big
- Slow
- Power-hungry
- Insecure

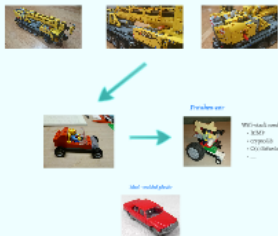
Linux 0.11 system
ran in 2MB

More features since then

Streamlining Linux

Modularity/Legos

Lego Model of Software



Subtractive Engineering

Folly of subtractive engineering



As system scales up, it's harder to remove than build from scratch

Nobody wants to remove stuff they don't understand



Just say "NO" to subtractive engineering

Lose community

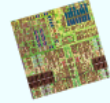
If we slim down Linux, it's not Linux anymore

No network effects

What is...
People want to leverage...

- Network stack
- File systems
- USB
- NFC

SOC support



Modularity/Legos

Lego Model of Software



Franken-car



Wifi-stack needs:

- ICMP
- crypto lib
- O(1) Scheduler
-

Ideal - molded plastic

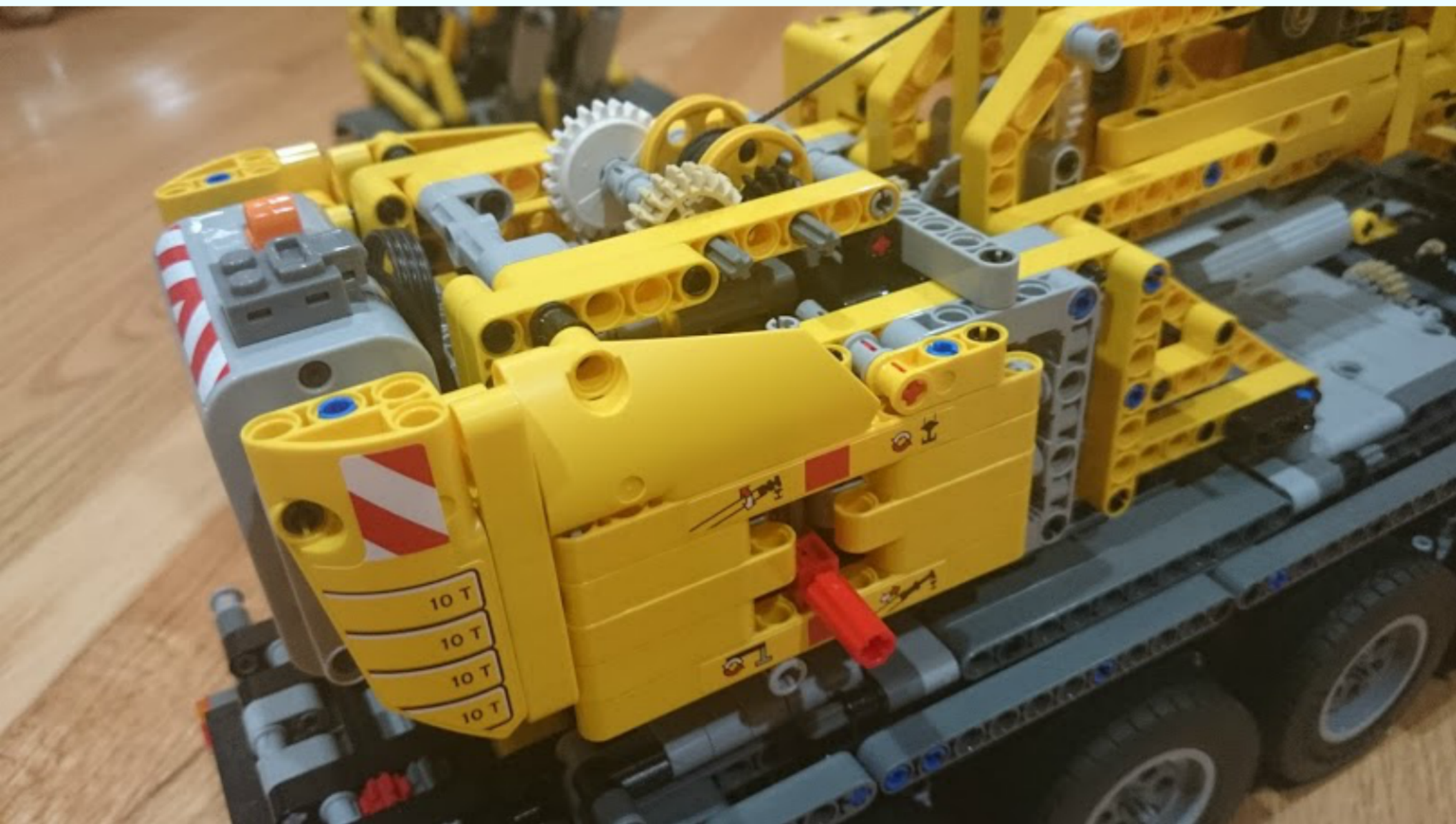


Su

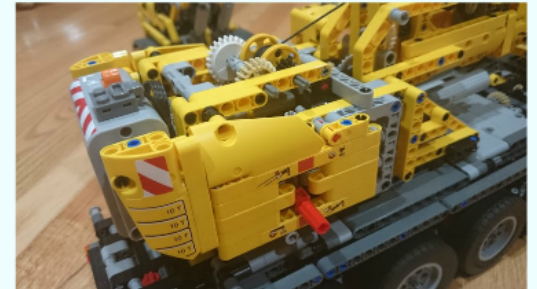
Folly of sub







Lego Model of Software

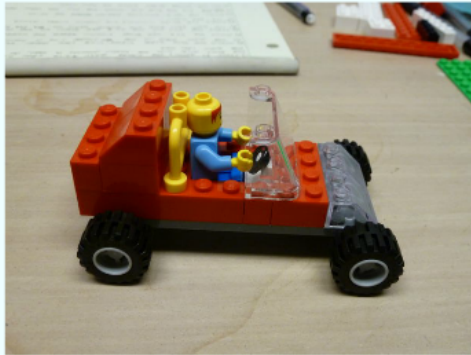


Franken-car



Wifi-stack needs:

- ICMP
- crypto lib
- $O(1)$ Scheduler
-



Franken-car



- Wifi-stack needs:
- ICMP
 - crypto lib
 - O(1) Scheduler
 -

Ideal - molded plastic





Franken-car



Wifi-stack needs:
• ICMP
• crypto lib
• O(1) Scheduler
• ...

ed plastic



If we slim d
it's not Lin

No netwo

Subtractive Engineering

Folly of subtractive engineering



As system scales up, it's harder to remove than build from scratch

Nobody wants to remove stuff they don't understand



Just say "NO" to subtractive engineering



Lose community

If we slim down Linux,
it's not Linux anymore

No network effects

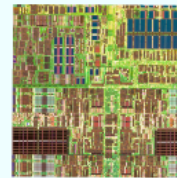
Reuse

What re-use are we striving for?

People want to leverage:

- Network stack
- File systems
- USB
- NFC

SOC support



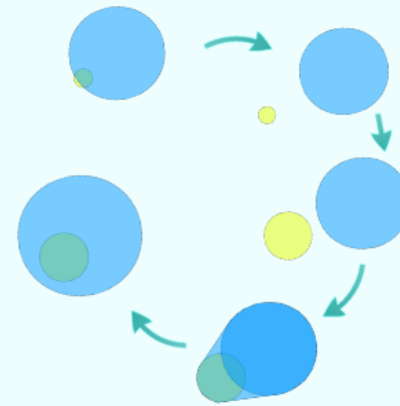
What to do?



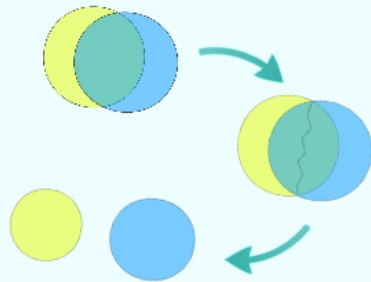
Fork!

Fragmentation

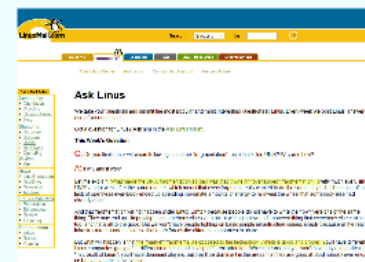
Good Fragmentation



Bad Fragmentation

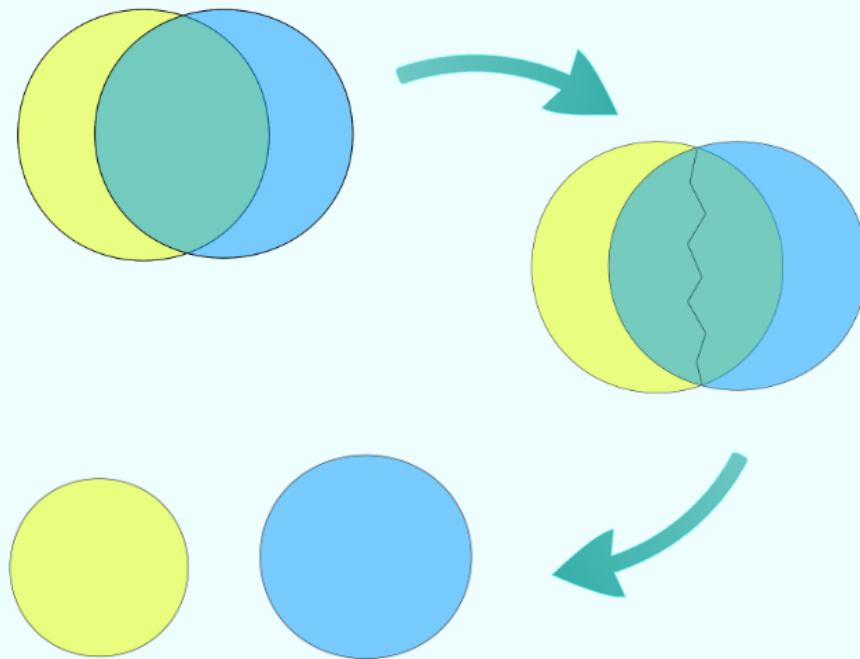


Linus Quote

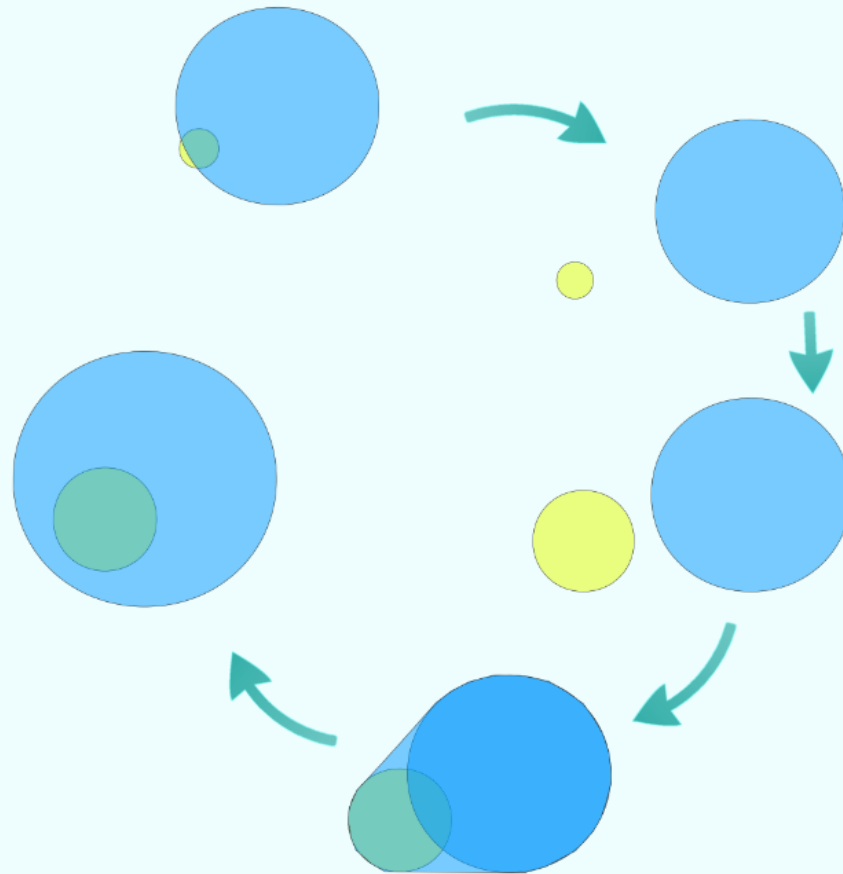


What to do?
Fork!

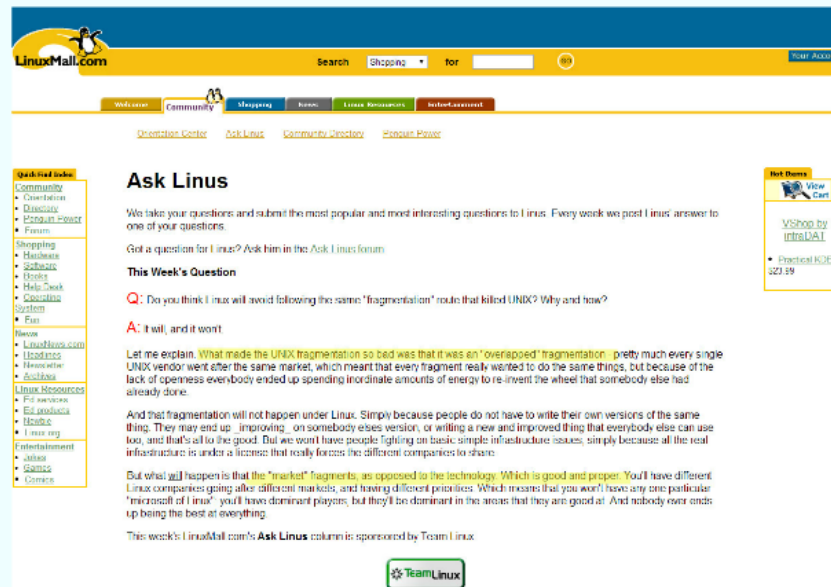
Bad Fragmentation



Good Fragmentation



Linus Quote

A screenshot of the LinuxMall.com website from May 2000. The page features a navigation bar with links like 'Web Store', 'Community', 'Shopping', 'Home', 'Linux Resources', and 'Help/Contact Us'. The main content area is titled 'Ask Linus' and contains a question and answer about UNIX fragmentation. The sidebar on the left lists various categories like 'Community', 'Shopping', 'Hardware', 'Software', 'Books', 'Help Desk', 'Cool Linux System', 'Fun', 'News', 'LinuxMall.com', 'Hardware', 'Software', 'Books', 'Linux Resources', 'FAQs', 'Entertainment', 'Links', and 'Contact'. The sidebar on the right has a 'Hot Items' section with a 'View Cart' button and a 'VShop by infraUI' button. The footer includes a 'TeamLinux' logo.

Linux Mall - Ask Linus, May 2000

We take your questions and submit the most popular and most interesting questions to Linus. Every week we post Linus' answer to one of your questions.

Got a question for Linus? Ask him in the [Ask Linus forum](#).

This Week's Question

Q: Do you think Linux will avoid following the same "fragmentation" route that killed UNIX? Why and how?

A: It will, and it won't.

Let me explain. What made the UNIX fragmentation so bad was that it was an "overlapped" fragmentation - pretty much every single UNIX vendor went after the same market, which meant that every fragment really wanted to do the same things, but because of the lack of openness everybody ended up spending inordinate amounts of energy to re-invent the wheel that somebody else had already done.

And that fragmentation will not happen under Linux. Simply because people do not have to write their own versions of the same thing. They may end up improving on somebody else's version, or writing a new and improved thing that everybody else can use too, and that's all to the good. But we won't have people fighting on basic simple infrastructure issues, simply because all the real infrastructure is under a license that really forces the different companies to share.

But what will happen is that the "market" fragments, as opposed to the technology. Which is good and proper. You'll have different Linux companies going after different markets, and having different priorities. Which means that you won't have any one particular "microsoft of Linux": you'll have dominant players, but they'll be dominant in the areas that they are good at. And nobody ever ends up being the best at everything.

This week's LinuxMall.com's **Ask Linus** column is sponsored by Team Linux.



Linux Mall - Ask Linus, May 2000

Open Source

How it works



Developers publish their derivative software

System only works when people do more than publish

They need to contribute



Network Effects



Creates a community



Open Source Network

Other developers write software you use

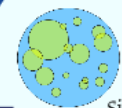
More developers

Ecosystem



Related services
Books, training
tools, jobs

Community size



- Not a single community
- Lots of sub-communities

Size matters

Some communities are very small



People are in multiple communities simultaneously

Cost to contribute

To build community, must create

Definition

Open Source Software can be freely used, changed and shared by anyone

Legal framework

Other Licenses

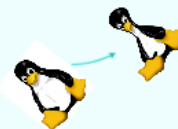
- BSD
- Apache
- MIT

Different Legal Terms

Re-Specialization



!urray's talk on Boot Time (Europe)



Embedded

Integrated function

camera
box

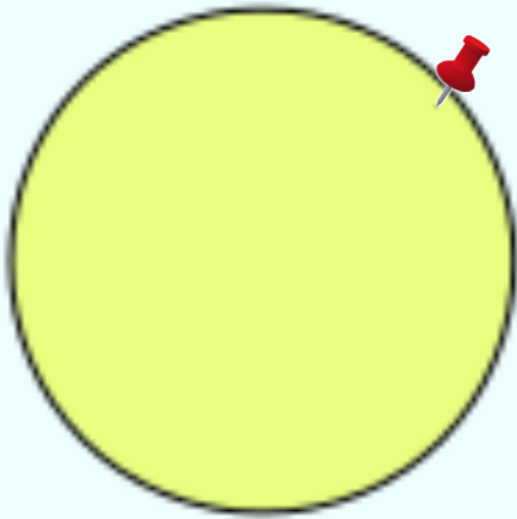
Mobile phones

between generalization
specialization

At the intersection of Open Source and Embedded



We need a new base camp
here!



ELCeption

- Pay attention to network effects
- Forking can equal growth

Thanks for your time