Best practices for long-term support and security of the device-tree

Alison Chaiken

Mentor Embedded Software

October 24, 2013

alison_chaiken@mentor.com

mentor embedded

mentor.com/embedded

Agenda

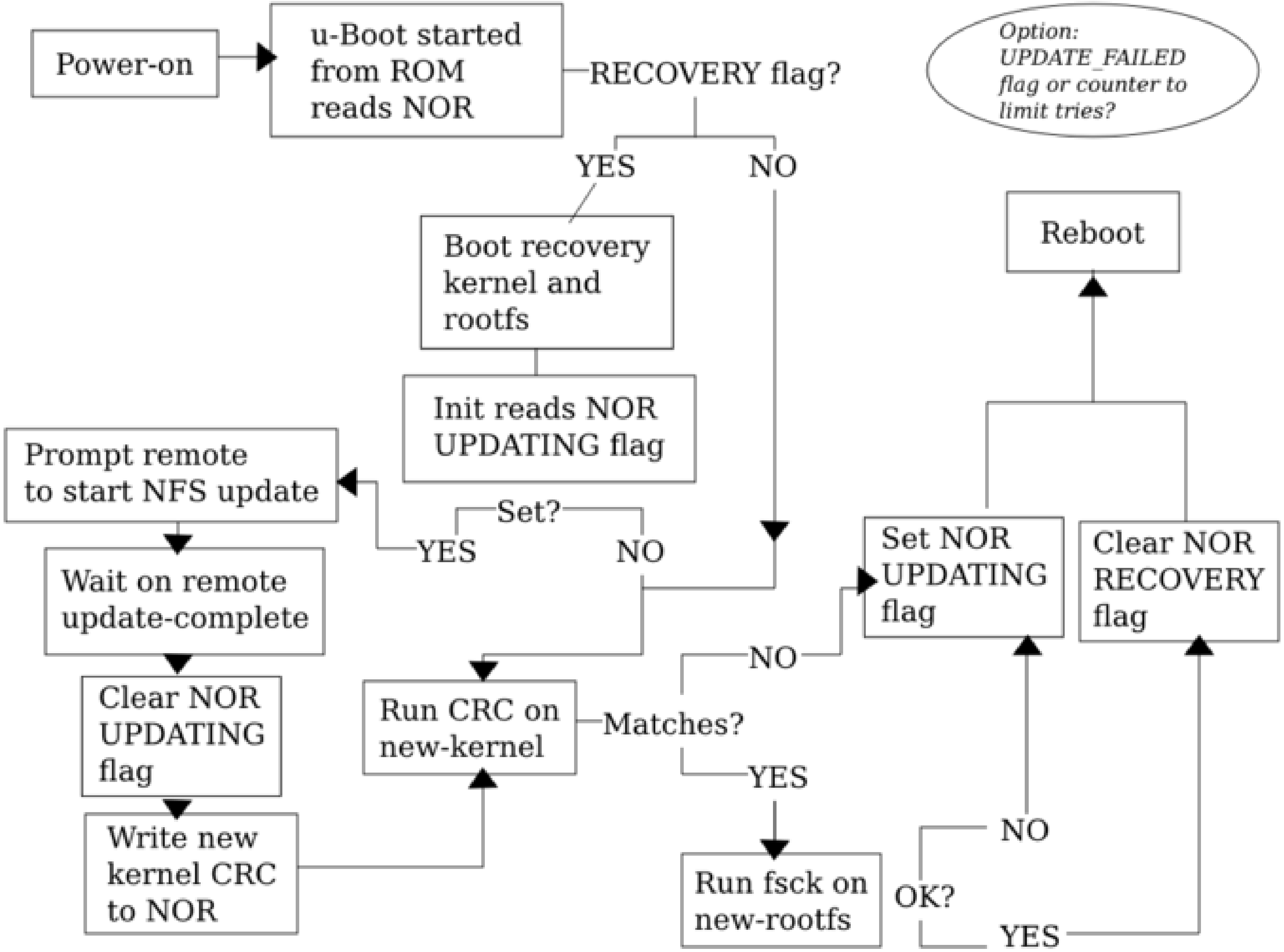
- How DT complicates updates
- Strategies for updates including DTBs
- Best practices in creating DTS
- Tools to make DT comprehensible to non-experts
- “Coming attractions”

mentor.com/embedded

2005 Model Year car is not easily updated

2015MY car will get new kernels;
also updated device-trees?

Updates are already tricky without DTB


Power-on
u-Boot started from ROM reads NOR
RECOVERY flag?
YES
NO
Option: UPDATE_FAILED flag or counter to limit tries?
Boot recovery kernel and rootfs
Reboot
Init reads NOR UPDATING flag
Prompt remote to start NFS update
Set?
YES
NO
Set NOR UPDATING flag
Clear NOR RECOVERY flag
Wait on remote update-complete
NO
Clear NOR UPDATING flag
Run CRC on new-kernel
Matches?
YES
Write new kernel CRC to NOR
NO
Run fsck on new-rootfs
OK?
YES

Consider security patches for Trusted Platform Module

- 2015MY car in 2025
  - one of 30 models that year, now 300 products ago
  - different by locality
- Needs a new kernel that stops the published exploit.
- Does it need a new device-tree too?
  - Maybe not if “device-tree is an ABI”?
  - But even so, 10 years later?
- How to update DTB?
- Updated DTB may need new shared-object libs or firmware.
- “Range anxiety” means power updates also likely.

Advocate changing DTB? No, but . . .

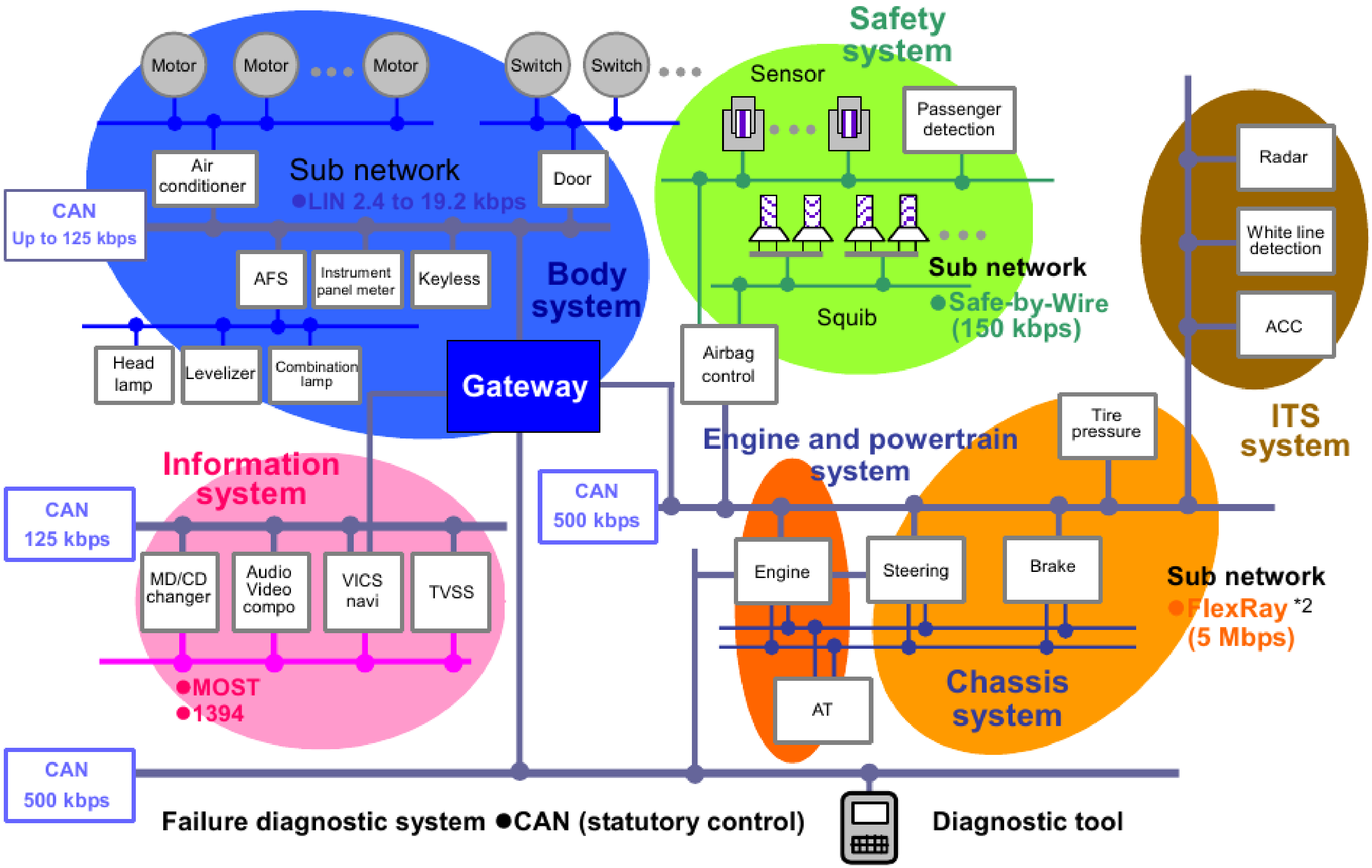
- Consider battery upgrades:

  many MCUs
- 2013MY: 5-year warranty

- DT's are supposed to be HW *description*
- Consider the amount of *configuration* already in DT's: MTD partition tables; boot device selection; allocation of oversubscribed resources

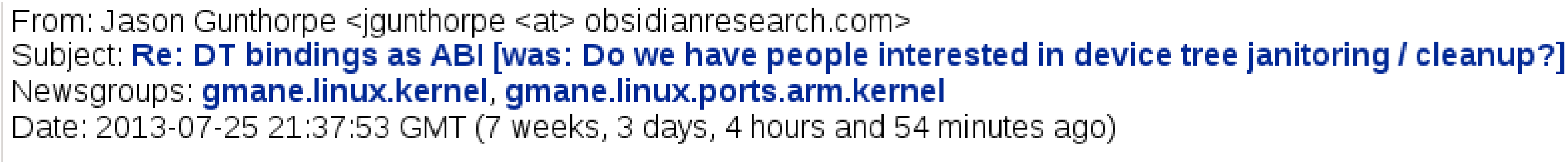
Why DTB updates are hard

Multiple processors in a LAN may need update

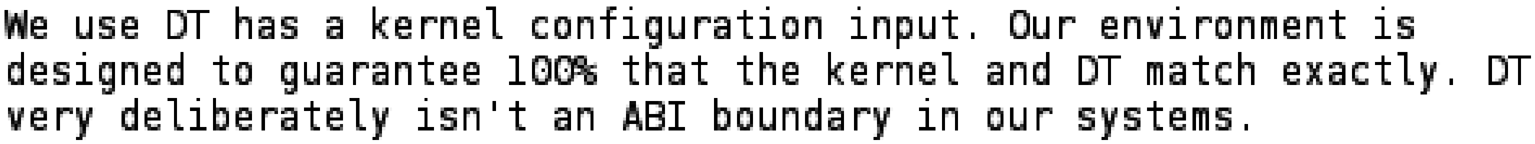


Copyright Renesas, "*Introduction to CAN*", with permission.

Careful: DTS and Kconfig must be compatible



From: Jason Gunthorpe <jgunthorpe <at> obsidianresearch.com>
Subject: Re: DT bindings as ABI [was: Do we have people interested in device tree janitoring / cleanup?]
Newsgroups: gmane.linux.kernel, gmane.linux.ports.arm.kernel
Date: 2013-07-25 21:37:53 GMT (7 weeks, 3 days, 4 hours and 54 minutes ago)



```
We use DT has a kernel configuration input. Our environment is
designed to guarantee 100% that the kernel and DT match exactly. DT
very deliberately isn't an ABI boundary in our systems.
```

Unintentionally changing drivers

- *Obvious:* device driver behavior is modified every time its DT node is.
- *Not so obvious:* A device driver is modified every time its clock, pinmux or power supply voltage are, perhaps by *another driver's* DT node.
- Koen Kooi's example:
  - “IMPORTANT: booting the existing am335x-bone.dts will blow up the HDMI transceiver after a dozen boots with an uSD card inserted because LDO will be at 3.3V instead of 1.8.   MMC support for AM335x still isn't in . . .”
- Scary for long-term support.

Three possible solutions

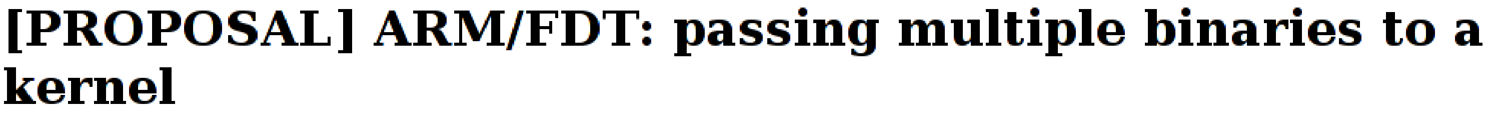
Fernandes @ ELC2013: Flattened Image Tree

- Addresses many of the toughest LTS problems.
- .its files record versions of DTS, kernel, misc. files via CRC.
  - Alternatively, use Bitbake recipe or Android-like manifest
- Enables OTA with one transferred archive.
- ChromeOS “Verified Boot” provides extensions for signed images.
- ChromeOS' Simon Glass added support for signing to Das U-boot.

DTS runtime configurability via Overlays (P. Antoniou)

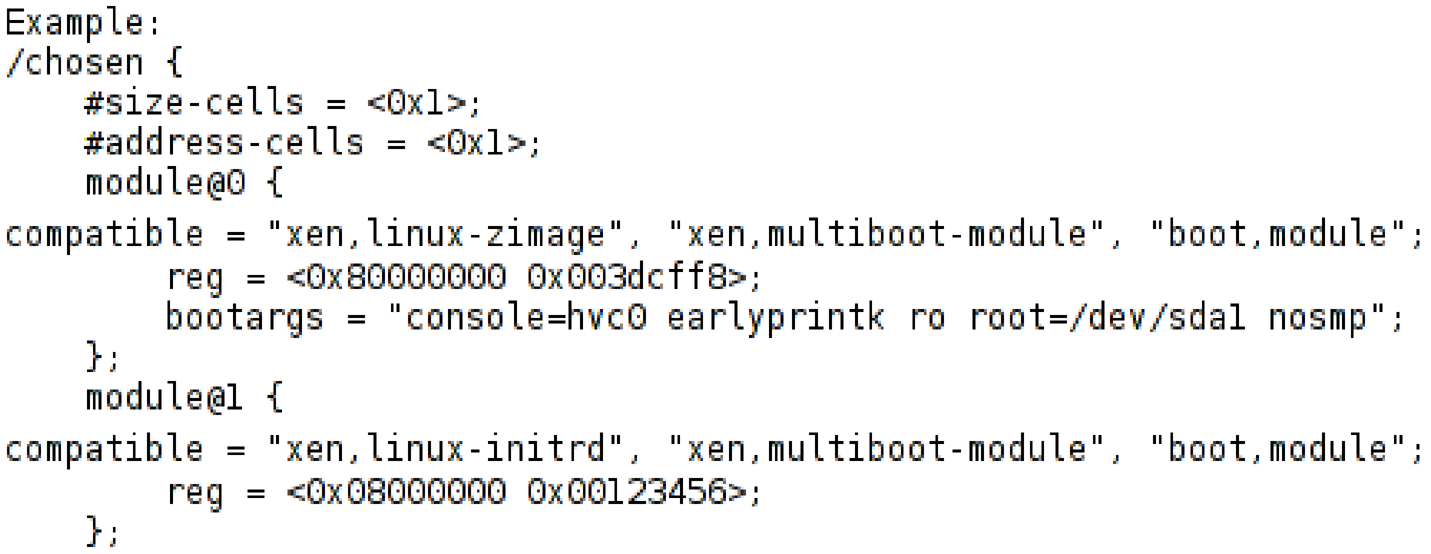
- Overlays, like udev, implement hot-pluggability.
  - Intended for daughter-cards.
- Overlay hardware can be added at runtime.
  - but only DTB fragments compiled along with dependencies from original DTS.
  - A limitation but a security feature, too.
- *Proposal:* use overlays as a method to update DTBs.
  - Not suitable for security improvements or boot devices.
  - Similar in spirit to unionfs in Knoppix.
- See Pantelis' talk and “Beaglebone and the 3.8 Kernel” paper.

Hypervisors in device-tree



[PROPOSAL] ARM/FDT: passing multiple binaries to a kernel

From: Andre Przywara *Date*: Tue, 03 Sep 2013



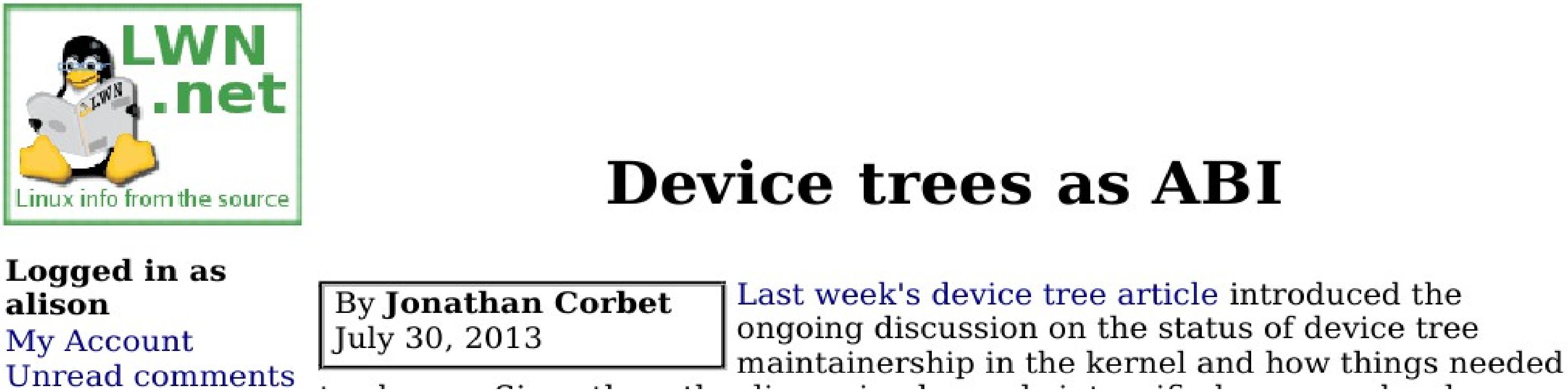
```
Example:
/chosen {
    #size-cells = <0x1>;
    #address-cells = <0x1>;
    module@0 {
compatible = "xen,linux-zimage", "xen,multiboot-module", "boot,module";
        reg = <0x80000000 0x003dcff8>;
        bootargs = "console=hvc0 earlyprintk ro root=/dev/sda1 nosmp";
    };
    module@1 {
compatible = "xen,linux-initrd", "xen,multiboot-module", "boot,module";
        reg = <0x08000000 0x00123456>;
    };
```

Another path to safe updates?

Making DTS more maintainable

Implications of “Device trees as ABI”

- Export all bindings to header file?
- Put dt.gz in /boot along with config files?
  - Or friendly lshw-like tool to present /proc DT state.
- Warn about unstable bindings with dtc or require explicit dtc switch to include unstable bindings?

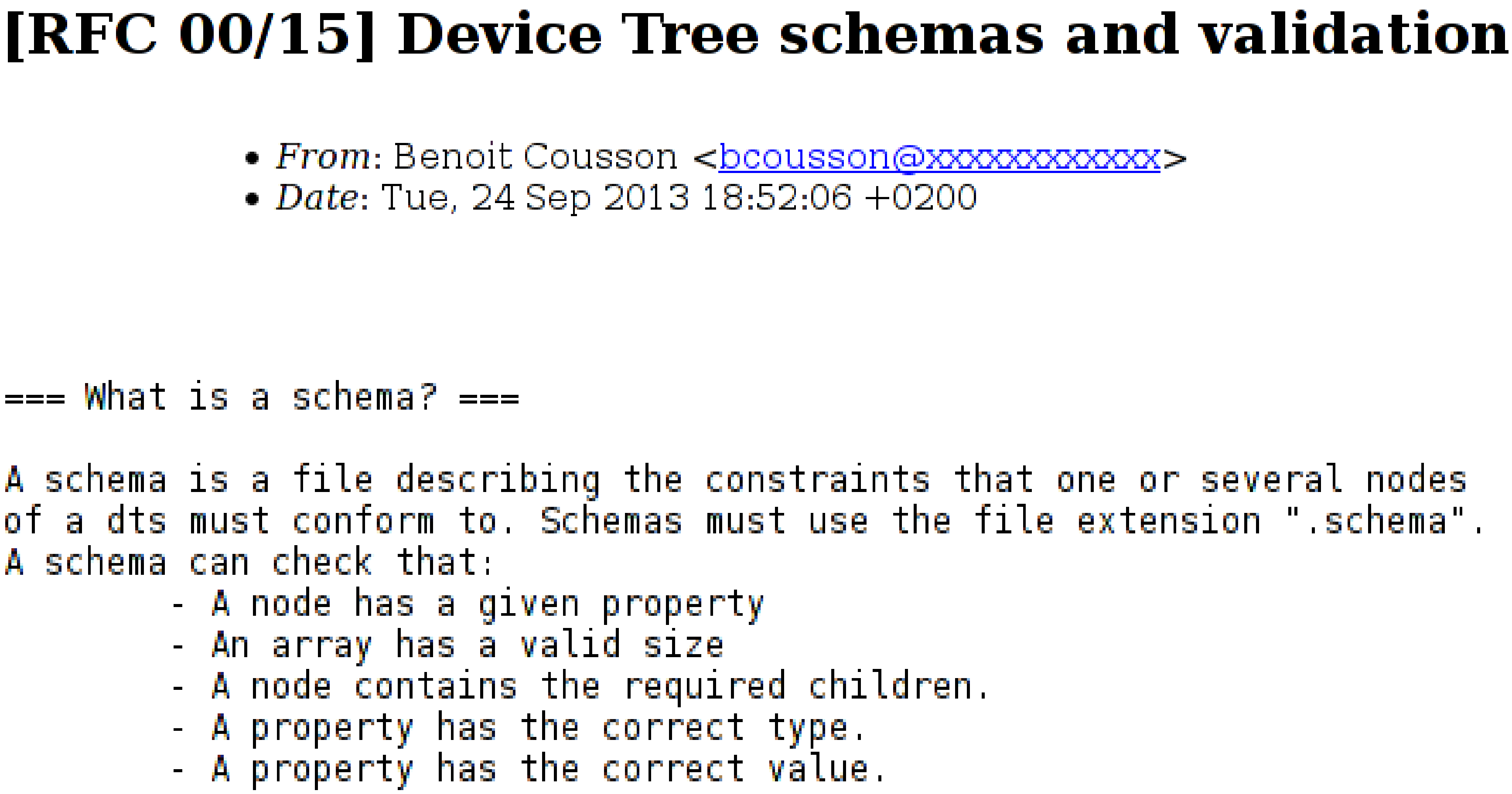


Device trees as ABI

Logged in as alison
My Account
Unread comments

By **Jonathan Corbet**
July 30, 2013

Last week's device tree article introduced the ongoing discussion on the status of device tree maintainership in the kernel and how things needed



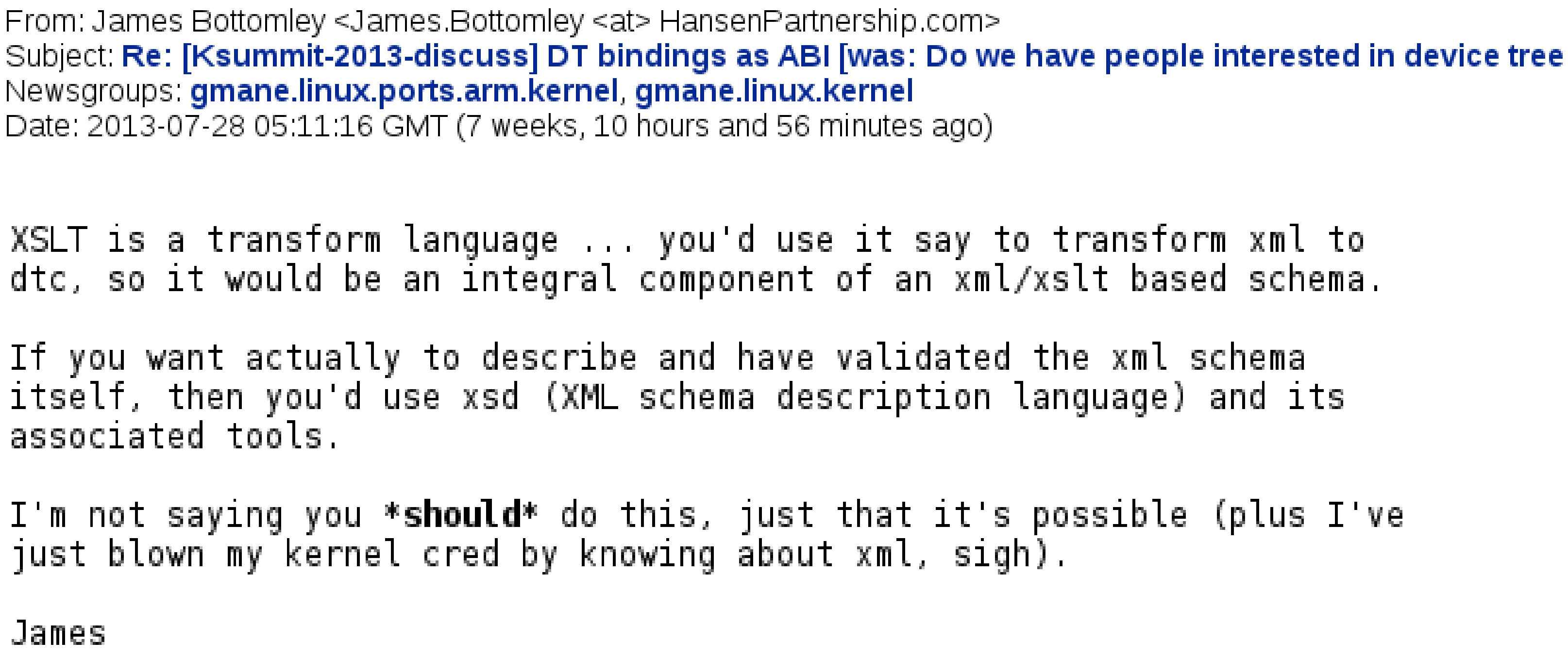
[RFC 00/15] Device Tree schemas and validation

- *From*: Benoit Cousson <bcousson@xxxxxxxxxxxx>
- *Date*: Tue, 24 Sep 2013 18:52:06 +0200

```
=== What is a schema? ===

A schema is a file describing the constraints that one or several nodes
of a dts must conform to. Schemas must use the file extension ".schema".
A schema can check that:
        - A node has a given property
        - An array has a valid size
        - A node contains the required children.
        - A property has the correct type.
        - A property has the correct value.
```

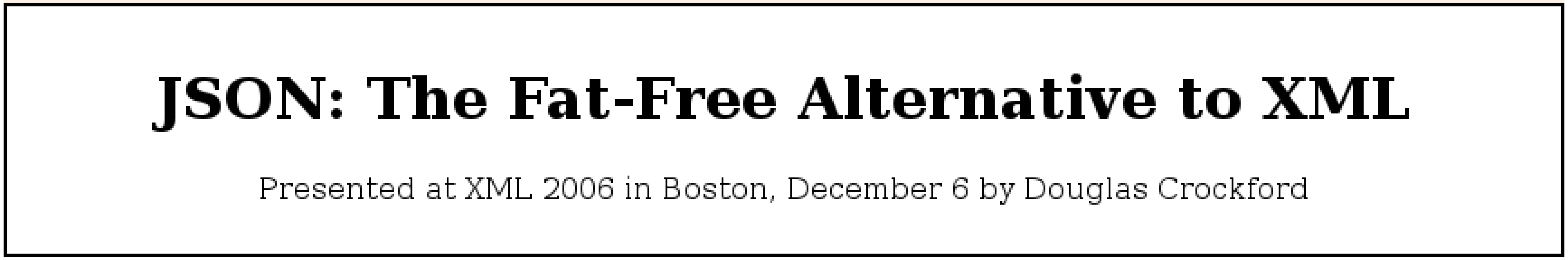
Consider XML for DT schema?

- Suggested on LKML by Arend van Spriel
- Lots of existing manipulation tools for XML.
- Reaction basically favorable:



From: James Bottomley <James.Bottomley <at> HansenPartnership.com>
Subject: **Re: [Ksummit-2013-discuss] DT bindings as ABI [was: Do we have people interested in device tree**
Newsgroups: **gmane.linux.ports.arm.kernel**, **gmane.linux.kernel**
Date: 2013-07-28 05:11:16 GMT (7 weeks, 10 hours and 56 minutes ago)

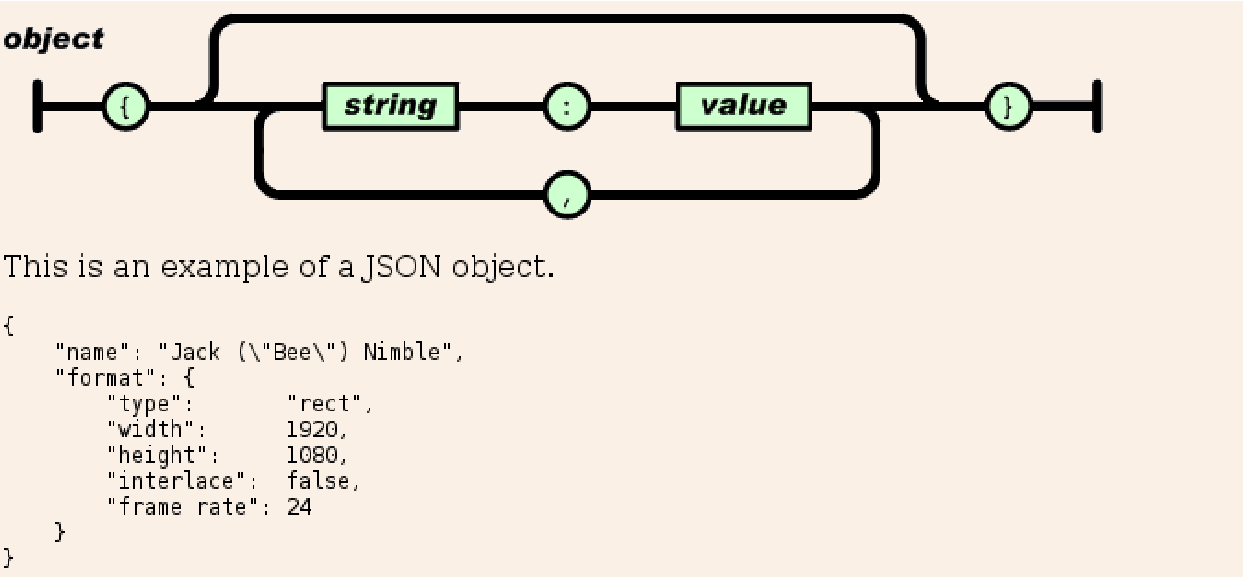
```
XSLT is a transform language ... you'd use it say to transform xml to
dtc, so it would be an integral component of an xml/xslt based schema.

If you want actually to describe and have validated the xml schema
itself, then you'd use xsd (XML schema description language) and its
associated tools.

I'm not saying you *should* do this, just that it's possible (plus I've
just blown my kernel cred by knowing about xml, sigh).

James
```



JSON: The Fat-Free Alternative to XML

Presented at XML 2006 in Boston, December 6 by Douglas Crockford

“JSON is a natural representation of data for the C family of programming languages.”



This is an example of a JSON object.

```
{
    "name": "Jack (\"Bee\") Nimble",
    "format": {
        "type":       "rect",
        "width":      1920,
        "height":     1080,
        "interlace":  false,
        "frame rate": 24
    }
}
```

"translated into Chinese, French, German, Italian, Japanese, Korean, and Spanish. . . . formalized in RFC 4627. . . . support for . . . C, C++, C#, . . . Erlang, Haskell, Java, . . . Lua, Perl, Objective-C, . . . PHP, Python,. . . Ruby, ."

***Proposal*: explicit inheritance in design**

- Create board.schema, cpu.schema, platdev.schema, <arch>.schema to represent base *classes*.
  - Each CPU nodewould need to validate against cpu.schema
  - each ARM platform against arm.schema,
  - each V4L2 camera device node against v4l2_capture.schema
  - Et cetera
- New DTC validator could enforce the inheritance for in-tree DTS.

Level separation

- Explicit inheritance in nesting of device-trees:

```
$ grep okay linux/arch/arm/boot/dts/*dtsi | wc
    181     732   11530
$ grep chosen linux/arch/arm/boot/dts/*.dtsi | wc
     22      80    1103
```

- Decisions about shared resources in low-level include files = premature optimization!
- Caution with status “okay”, “chosen,” “config-on-boot” and configuration of (power-supply, pinmux and clocks) at “leaf” level.

Tools: existing and desirable

- Extensions to DTC to support schemata validation.
- Verbose mode for DTC.
- Tools to answer the questions:
  - "In which DTSI file does a wrong value originate?"
  - "Do my Kconfig and DTS selections match?"
  - "Which of this huge set of DTS files can produce the DTB in the shipping product?"
- Support for signing DTBs which are not appended.
- Centralized registry for all "stable" and "staging" bindings.
- Graphviz/dotty extensions to visualize FDT.

What DTS output does preprocessor produce? *makedts* for ARM and x86

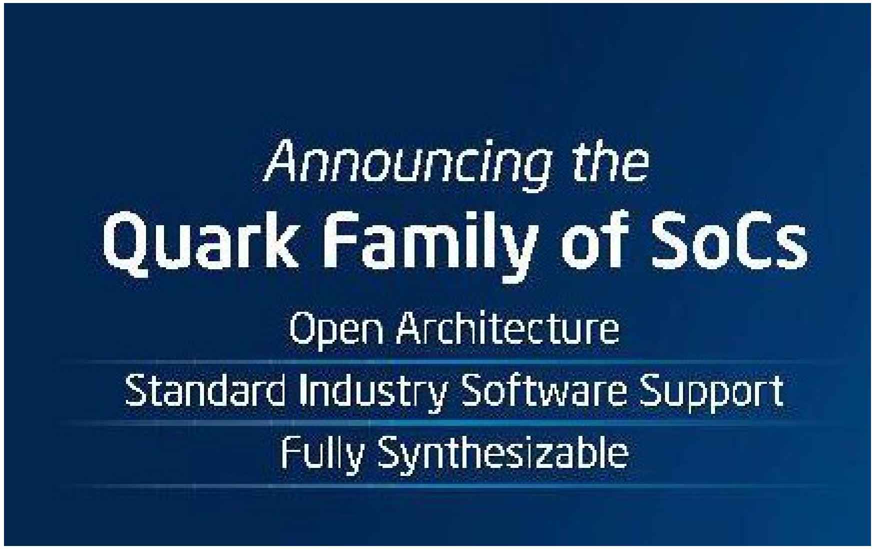
Advantage over fdtdump: outputs strings as characters, not ASCII codes.

Invoke with 'makedts <full-path-to-dts-file>' :

```
#!/bin/bash
DTC_CPP_FLAGS="-E -Wp,-MD,$BASE.pre.tmp -nostdinc /
 -Iarch/$SRCARCH/boot/dts -Iarch/$SRCARCH/boot/dts/include /
 -undef -D__DTS__  -x assembler-with-cpp"

#run C preprocessor
$CC $DTC_CPP_FLAGS -o $BASE.tmp $1

#run DTC
scripts/dtc/dtc -O dts -o $BASE.out.dts -b 0 -i arch/arm/boot/dts /
-d $BASE.dtc.tmp $BASE.tmp
```

For x86, first compile dtc itself.

x86 users will want DTS, too



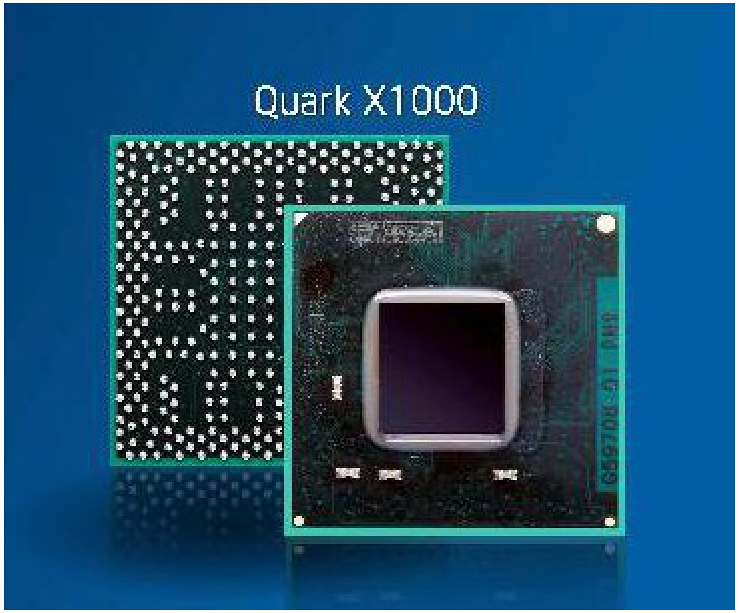
Announcing the

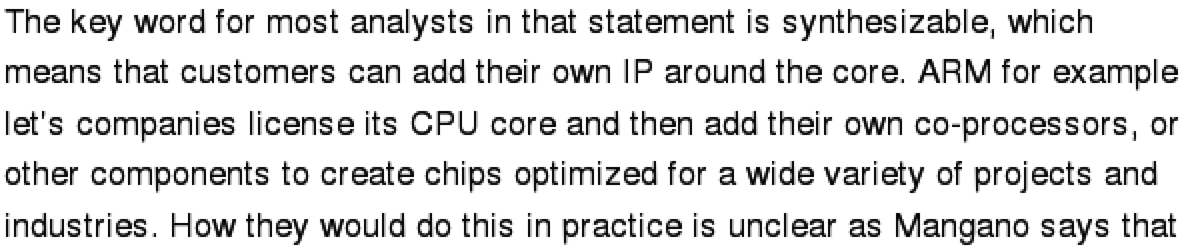
Quark Family of SoCs

Open Architecture

Standard Industry Software Support

Fully Synthesizable





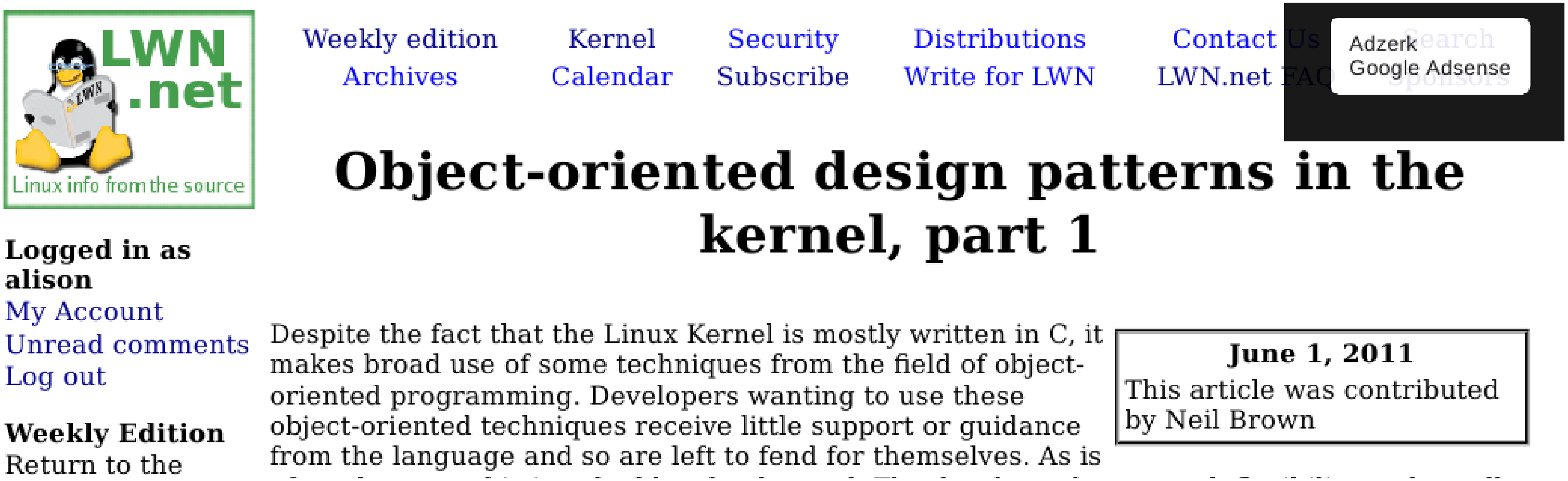
The key word for most analysts in that statement is synthesizable, which means that customers can add their own IP around the core. ARM for example let's companies license its CPU core and then add their own co-processors, or other components to create chips optimized for a wide variety of projects and industries. How they would do this in practice is unclear as Mangano says that

GigaOm, 9/10/2013

Summary: Best practice candidates

- Preserving *sets* of {bootloader configs, DTS sets, Kconfig and kernel sources}
  - FIT? BB recipes? Android-style “repo” manifests?
- Validation of device-trees:
  - DTC-based via new schemas and Warren's binding checklist.
  - CRCs via bootloader
- Design of *maintainable* device-trees via
  - Level separation
  - Overlays and conditionals rather than DTS proliferation.
- Preserve failsafe device-tree
  - Updates via overlay and into hypervisor domains

Conclusions



Object-oriented design patterns in the kernel, part 1

June 1, 2011

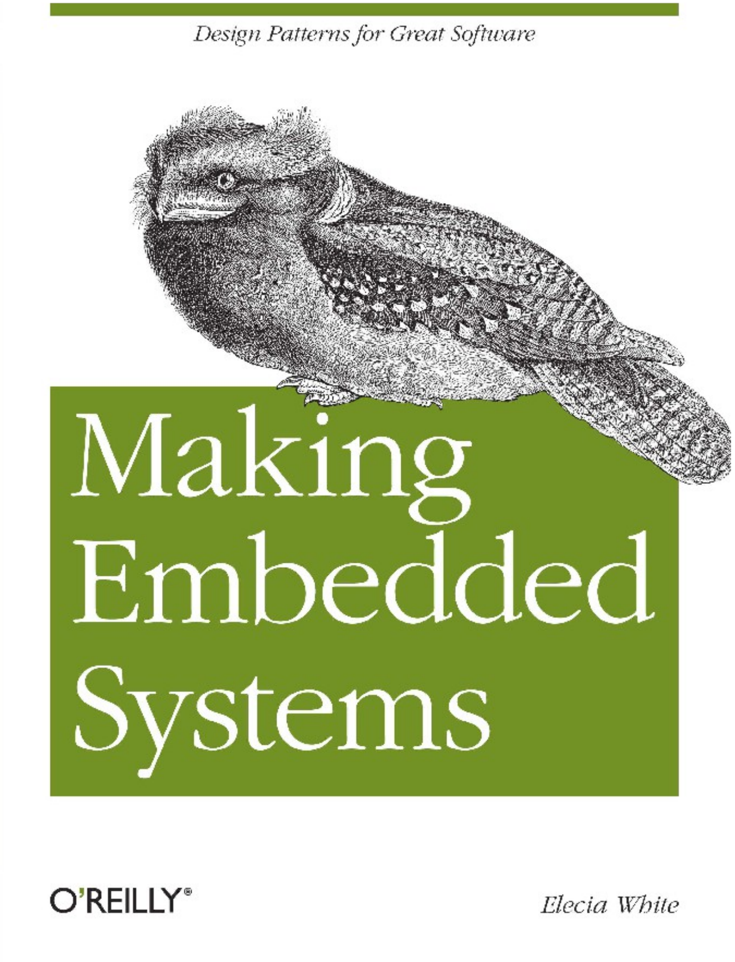
This article was contributed by Neil Brown

Despite the fact that the Linux Kernel is mostly written in C, it makes broad use of some techniques from the field of object-oriented programming. Developers wanting to use these object-oriented techniques receive little support or guidance from the language and so are left to fend for themselves. As is

- DT represents application of SW best practices to the kernel.
- Much criticized, but change is hard!

Following slides are extras

A book about design patterns for embedded



Uncle Bob Martin's 5 Agile Design Principles

1. Dependency Inversion Principle
2. Interface Segregation Principle
3. Liskov Substitution Principle
4. Open-Closed Principle
5. Single Responsibility Principle

From 1998, C++ Report, available at objectmentor.com