The new hwmon device registration API

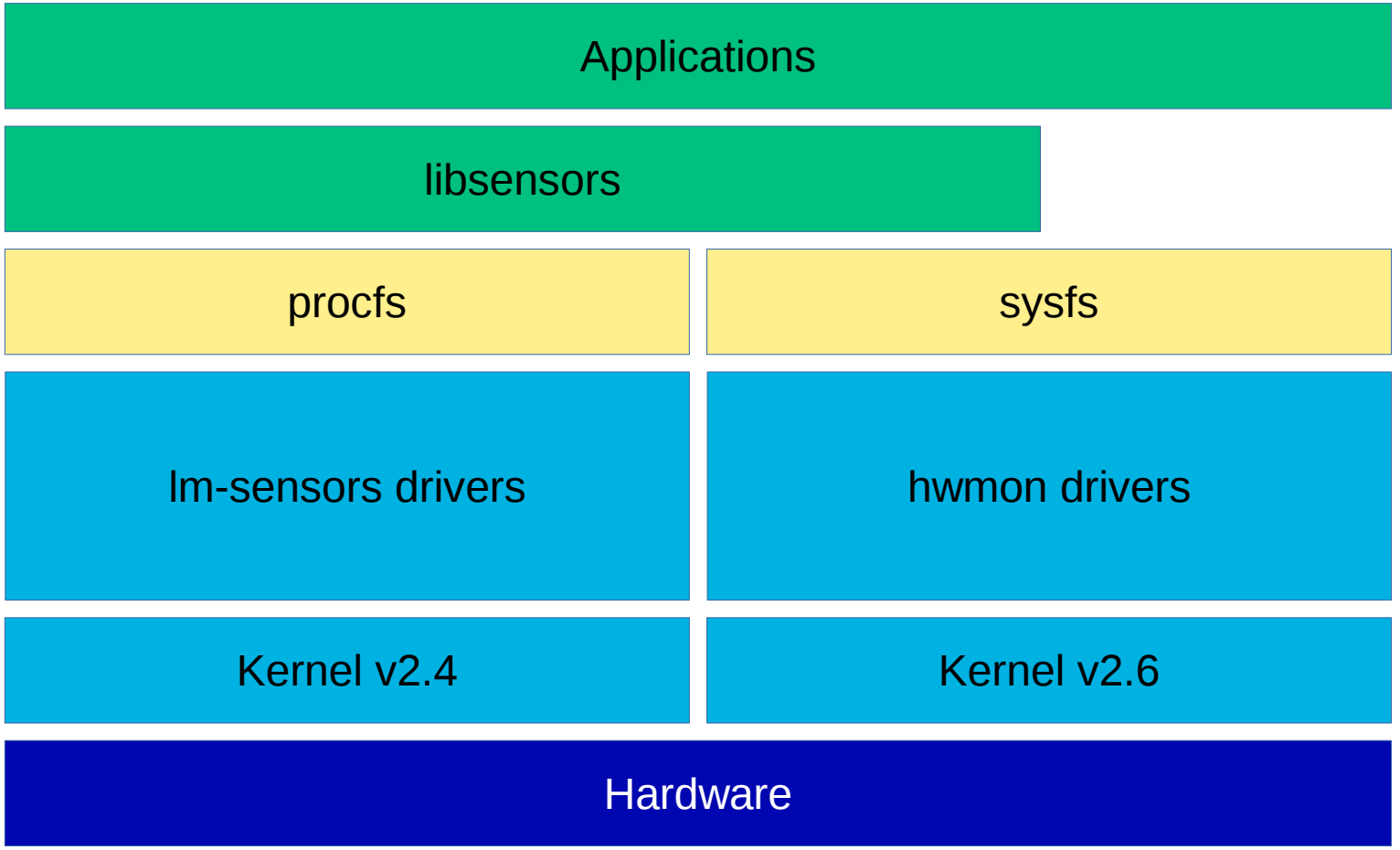
Jean DELVARE
Level 3 Support Engineer
SUSE

Origins of the hwmon subsystem

- 1998: 2 devices, 2 procfs-based drivers (lm75 and lm78)
- 1999: lm-sensors project
- Audience: developers and hobbyists
- 2002-2003: merged into kernel v2.5, sysfs-based
- 2006: 88 devices, 57 drivers
- 1906 device-specific defines in library
- Horrible and unmaintainable user-space

Chapter 1: user-space

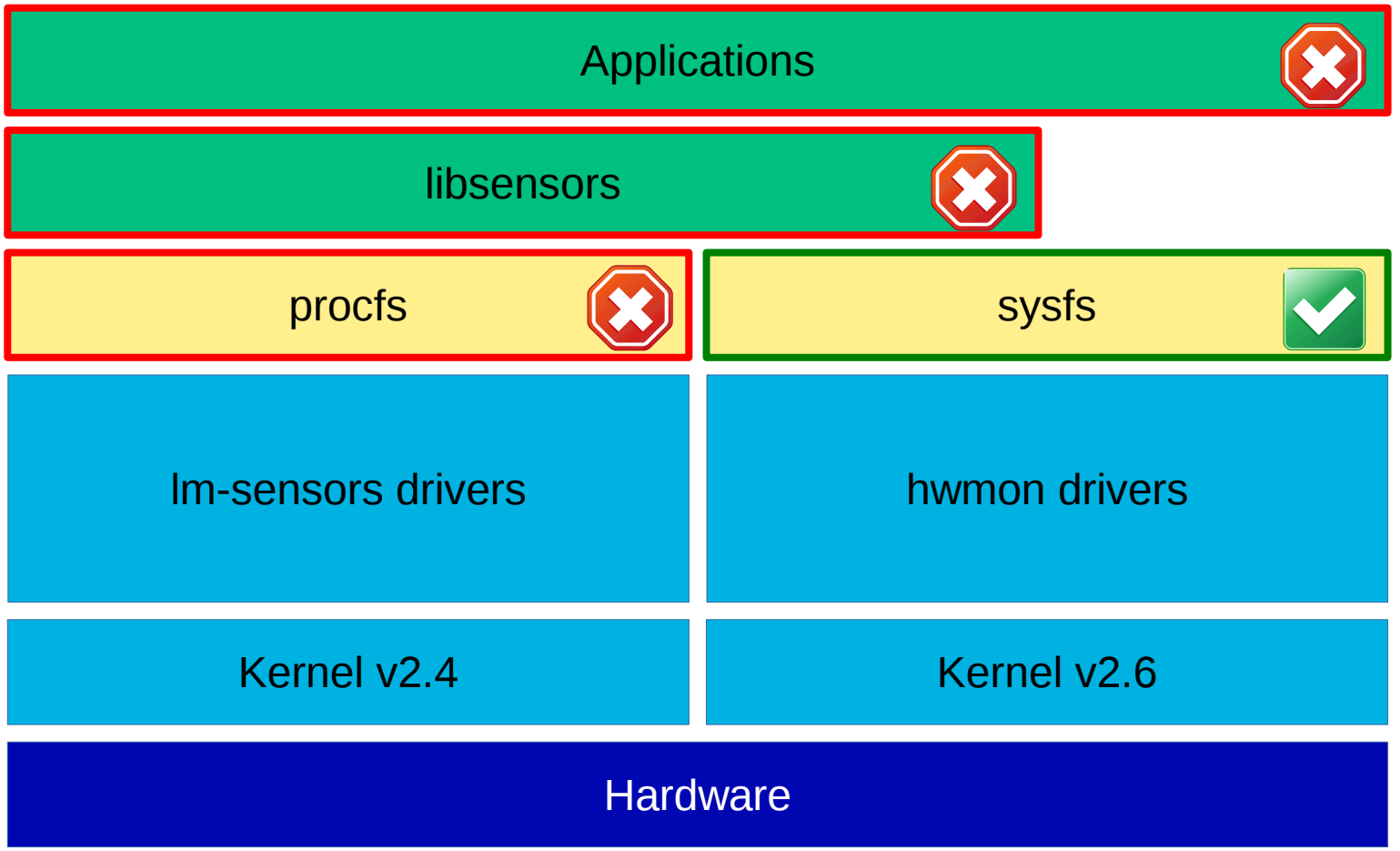
lm-sensors v2 in 2004



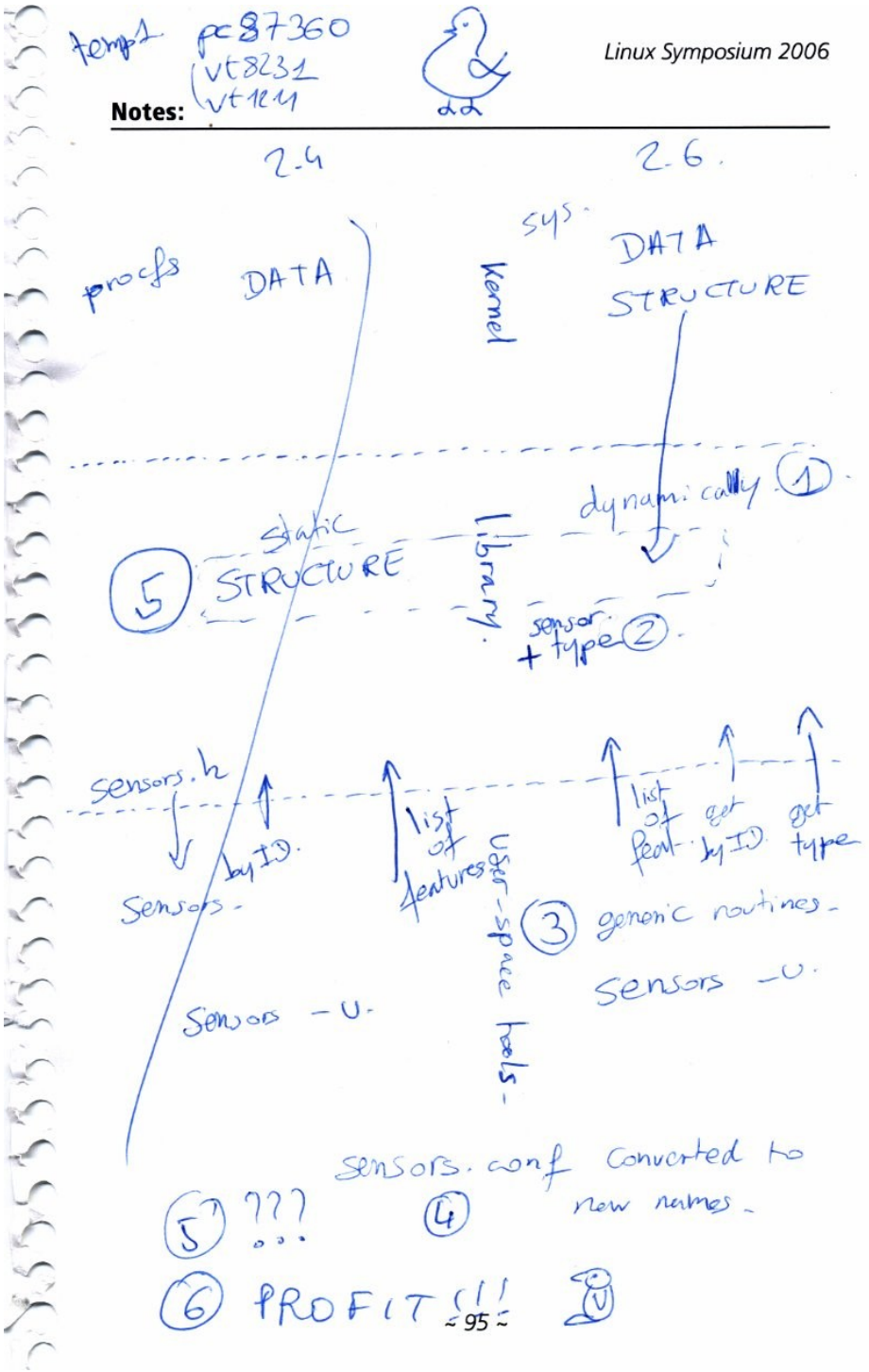
2006 problems

- No standard procfs interface
- Adding support for a new device required:
  - New driver or new code in existing driver
  - New code in (huge) libsensors
  - New code in every monitoring application
- Inconsistencies all over the place
- Very easy to introduce bugs

lm-sensors v2 in 2004



OLS 2006

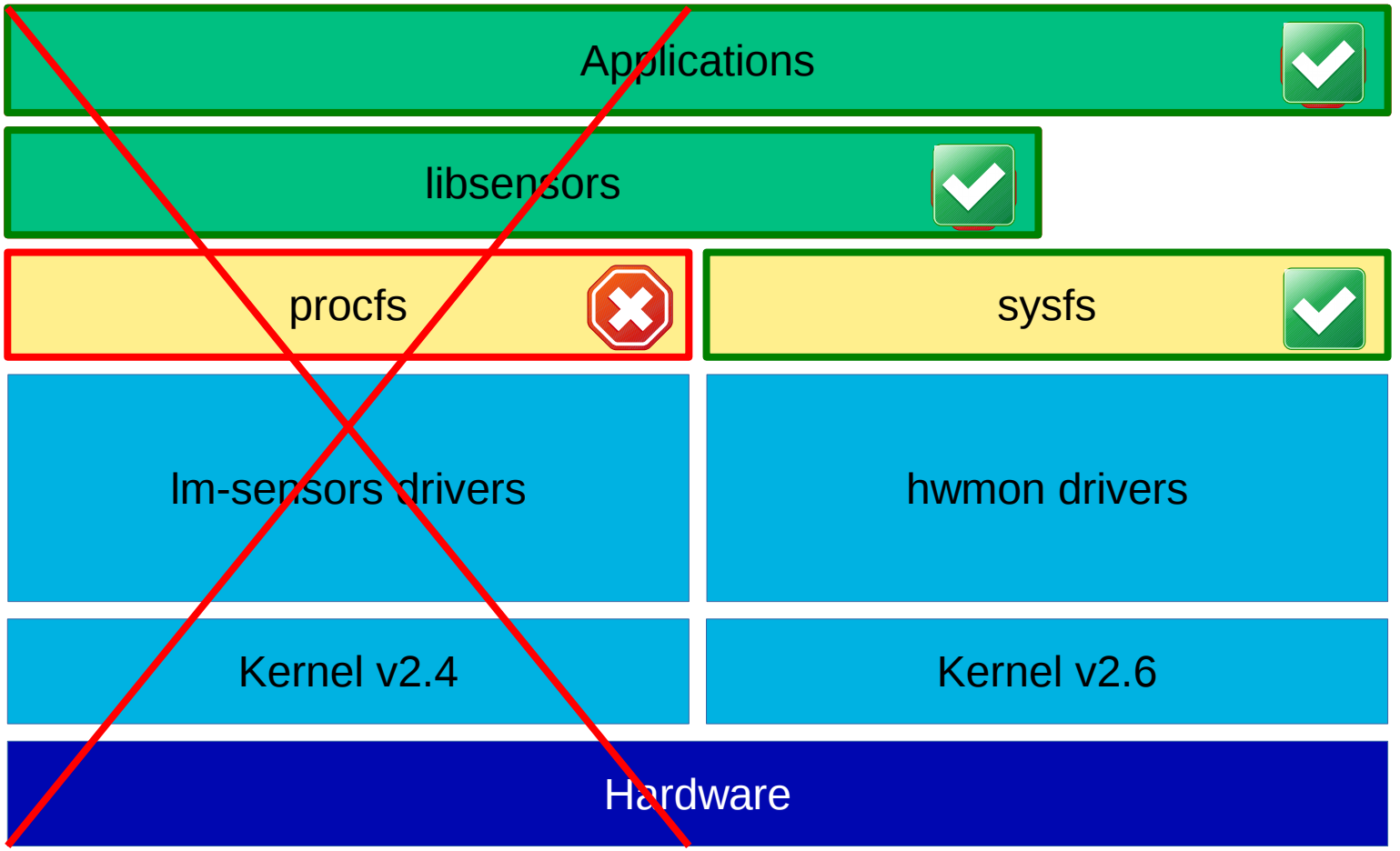

temp1
pc87360
vt8231
vt1211
Linux Symposium 2006
Notes:
2.4
2.6
procfs
DATA
sys
kernel
DATA
STRUCTURE
dynamically
1
static
STRUCTURE
5
library
sensor
+ type
2
sensors.h
by ID
Sensors -
Sensors - U.
list
of
features
User-space tools -
list
of
feat
get
by ID
get
type
3
generic routines -
sensors - U.
sensors.conf converted to
new names -
5'
???
4
6
PROFIT!!!
~95~

lm-sensors v3

- Based on `Documentation/hwmon/sysfs-interface`
- Using the one-value-per-file sysfs rule
- Using standard names
- Using standard units

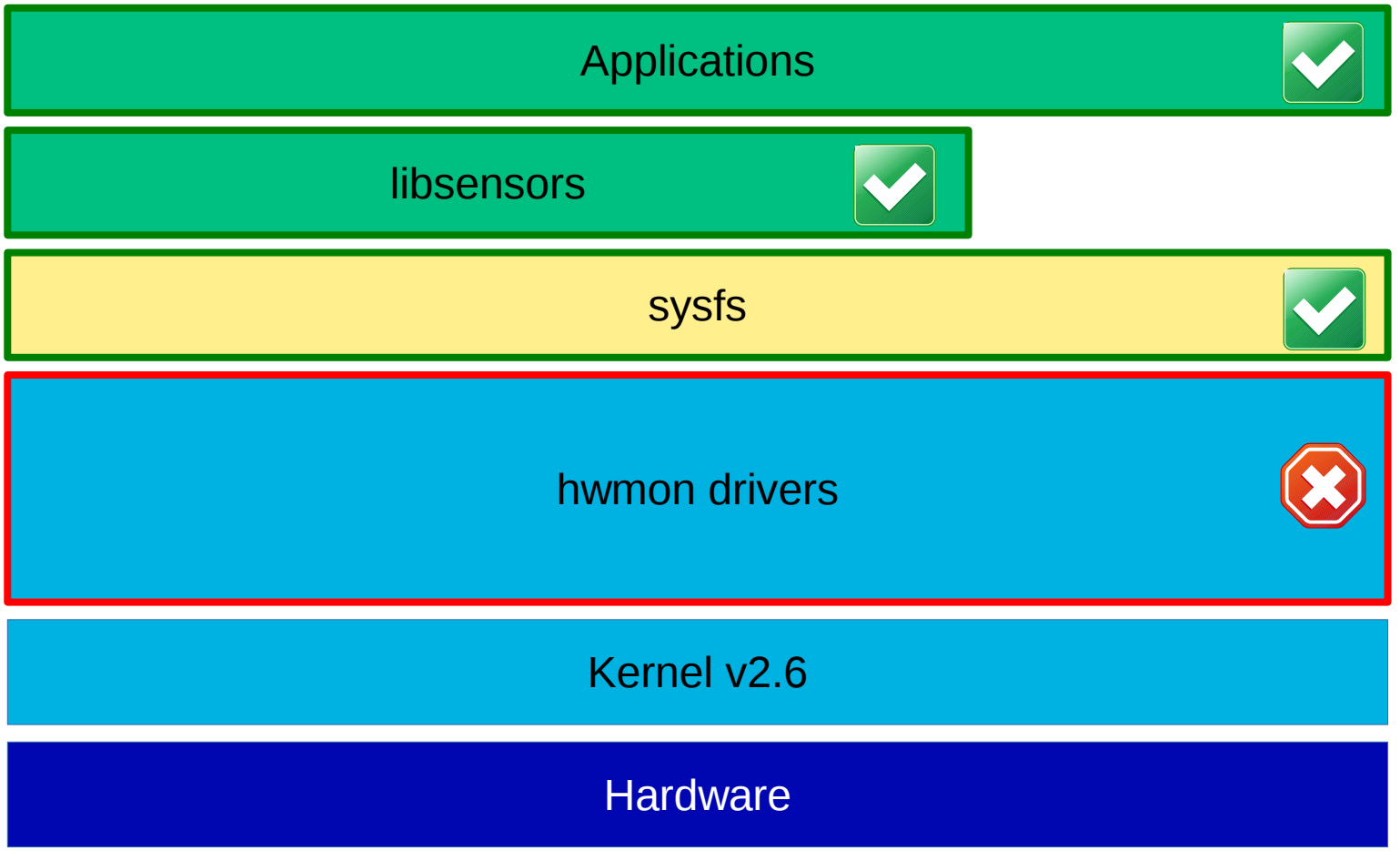
- No more device-specific code in libsensors
  - Library size down by 77%
- No more device-specific code in applications
  - “sensors” size down by 87%
- Support for new devices can be added without touching user-space
- Kernels v2.6.0 to v2.6.9: Inconsistencies detected and fixed

lm-sensors v3 in 2006



Chapter 2: kernel-space

lm-sensors v3 in 2005



July 2005: birth of the hwmon subsystem

```
commit 1236441f38b6a98caf4c7983e7efdecc2d1527b5
Author: Mark M. Hoffman <mhoffman@lightlink.com>
Date:   Fri Jul 15 21:38:08 2005 -0400

    [PATCH] I2C hwmon: hwmon sysfs class

    This patch adds the sysfs class "hwmon" for use by hardware monitoring
    (sensors) chip drivers.  It also fixes up the related Kconfig/Makefile
    bits.

    Signed-off-by: Mark M. Hoffman <mhoffman@lightlink.com>
    Signed-off-by: Jean Delvare <khali@linux-fr.org>
    Signed-off-by: Greg Kroah-Hartman <gregkh@suse.de>
```

July 2005 hwmon device registration API

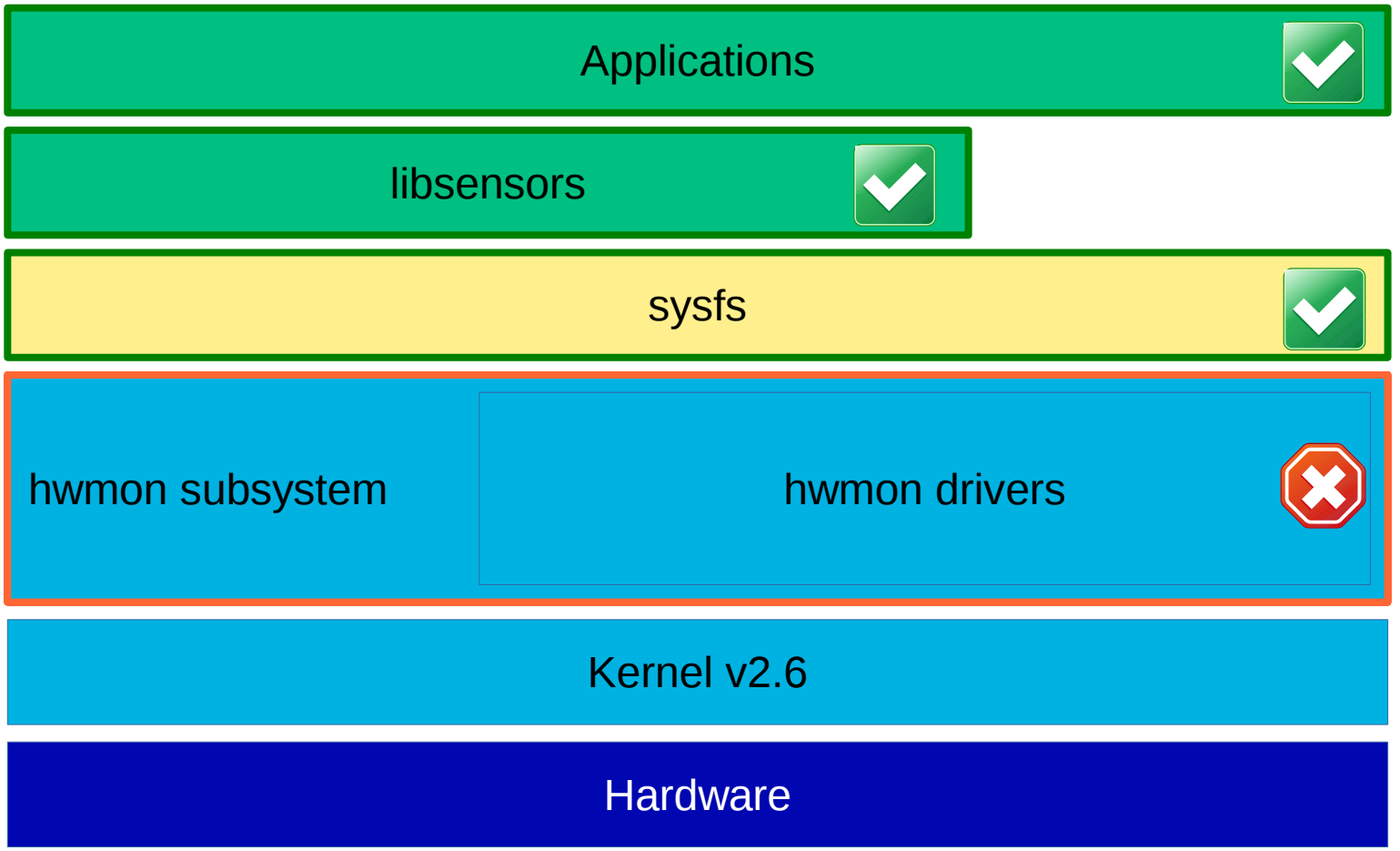
Minimalistic API in kernel v2.6.14:

```
struct class_device *hwmon_device_register(struct device *dev);

void hwmon_device_unregister(struct class_device *cdev);
```

Driver conversion: trivial.

lm-sensors v3 in 2006



July 2013: register devices with sysfs group

```
commit bab2243ce1897865e31ea6d59b0478391f51812b
Author: Guenter Roeck <linux@roeck-us.net>
Date:   Sat Jul 6 13:57:23 2013 -0700

    hwmon: Introduce hwmon_device_register_with_groups

    hwmon_device_register_with_groups() lets callers register a hwmon device
    together with all sysfs attributes in a single call.

    When using hwmon_device_register_with_groups(), hwmon attributes are
    attached to the hwmon device directly and no longer with its parent
    device.

    Signed-off-by: Guenter Roeck <linux@roeck-us.net>
```

July 2013 hwmon device registration API

Kernel v3.13:

```
struct device *
hwmon_device_register_with_groups(struct device *dev, const char name,
                                  void *drvdata,
                                  const struct attribute_group **groups);
```

Driver conversion: easy.

Group list gives flexibility. Special cases can be handled by .is_visible.

We can validate the device name.

As of today: 13/157 driver.

July 2013: managed device registration

```
commit 74188cba088192e14cd7fd5433876e8c947bcdd8
Author: Guenter Roeck <linux@roeck-us.net>
Date:   Thu Jul 11 20:00:12 2013 -0700

    hwmon: Provide managed hwmon registration

    Drivers using the new hwmon_device_register_with_groups API often have a
    remove function which consists solely of a call hwmon_device_unregister().

    Provide support for devm_hwmon_device_register_with_groups and
    devm_hwmon_device_unregister to allow this repeated code to be removed
    and help eliminate error handling code.

    Signed-off-by: Guenter Roeck <linux@roeck-us.net>
```

July 2013 hwmon device registration API

Kernel v3.13:

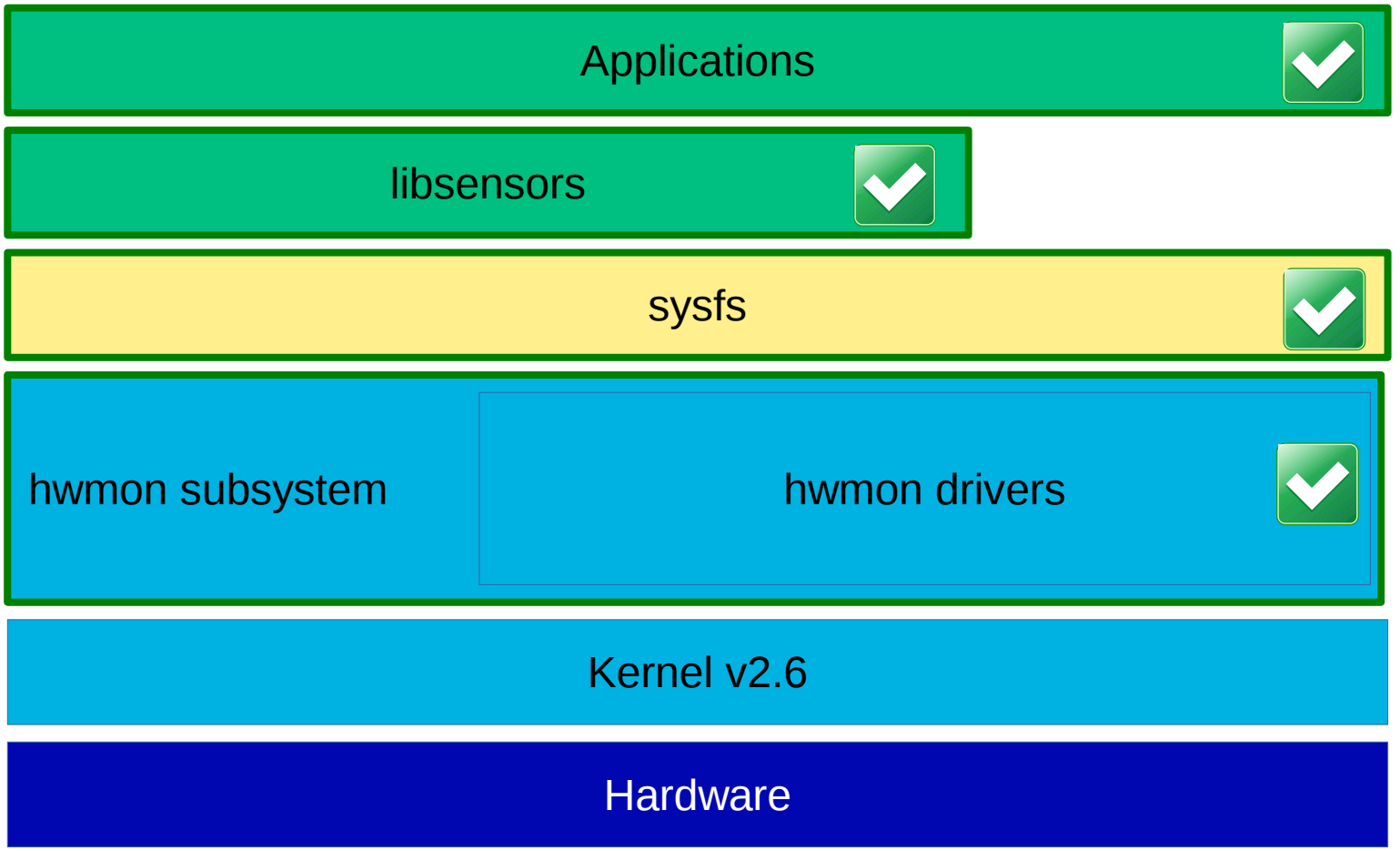
```
struct device *
devm_hwmon_device_register_with_groups(struct device *dev, onst char *name,
                                       void *drvdata,
                                       const struct attribute_group **groups);

void devm_hwmon_device_unregister(struct device *dev);
```

Driver conversion: trivial.

As of today: 90/157 driver.

lm-sensors v3 in 2013



July 2016: New hwmon registration API (prop.)

From: **Guenter Roeck** <linux@roeck-us.net>
Subject: [PATCH v3 2/**9**] **hwmon: (core) New hwmon registration API**
Date: Sun, 24 Jul 2016 20:32:25 -0700

Up to now, each hwmon driver has to implement its own sysfs attributes.
This requires a lot of template code, and distracts from the driver's core
function to read and write chip registers.

To be able to reduce driver complexity, move sensor attribute handling
and thermal zone registration into hwmon core. By using the new API,
driver code and data size is typically reduced by 20-70%, depending
on driver complexity and the number of sysfs attributes supported.

With this patch, the new API only supports thermal sensors. Support for
other sensor types will be added with subsequent patches.

Acked-by: Punit Agrawal <punit.agrawal@arm.com>
Reviewed-by: Jonathan Cameron <jic23@kernel.org>
Signed-off-by: Guenter Roeck <linux@roeck-us.net>

2017 hwmon device registration API?

Future kernel:

```
struct device *
hwmon_device_register_with_info(struct device *dev,
                const char *name, void *drvdata,
                const struct hwmon_chip_info *info,
                const struct attribute_group **groups);
struct device *
devm_hwmon_device_register_with_info(struct device *dev,
                      const char *name, void *drvdata,
                      const struct hwmon_chip_info *info,
                      const struct attribute_group **groups);
```

Driver conversion: needs actual work.

Still under review and discussion.

As of today: 9/157 driver pending review and merging.

Conclusions

A good subsystem should help drivers:

- Integrate into the kernel (device registration, resource management)
- Offer a standard interface

Benefits:

- Smaller binary size
- Fewer bugs
- Easier integration with thermal subsystem

Concerns:

- Performance
- Too many ways to register hwmon devices

Questions?


SUSE
We adapt. You succeed.