How to decide Linux Kernel for Embedded Products

Tsugikazu SHIBATA

NEC

20, Feb. 2013

Embedded Linux Conference 2013

Parc55 @ SAN FRANCISCO

Agenda

- Points to be considered to decide Linux kernel version

- Key activities of LTSI

Points to be considered to decide the kernel version

- ☑ Technical aspects
- ☑ Stability
- ☑ Maintenance
- ☑ Cost
- ☑ …

Technical aspects

- Performance
- Memory
- Battery life
- Real time nests
- Connect ability
- …

Technical aspects

- Performance
- Memory
- Battery life
- Real time nests
- Connect ability
- …

Technical aspects are heavily depends on kernel development in the community.
We should look at the community otherwise we cannot choose right version of the kernel

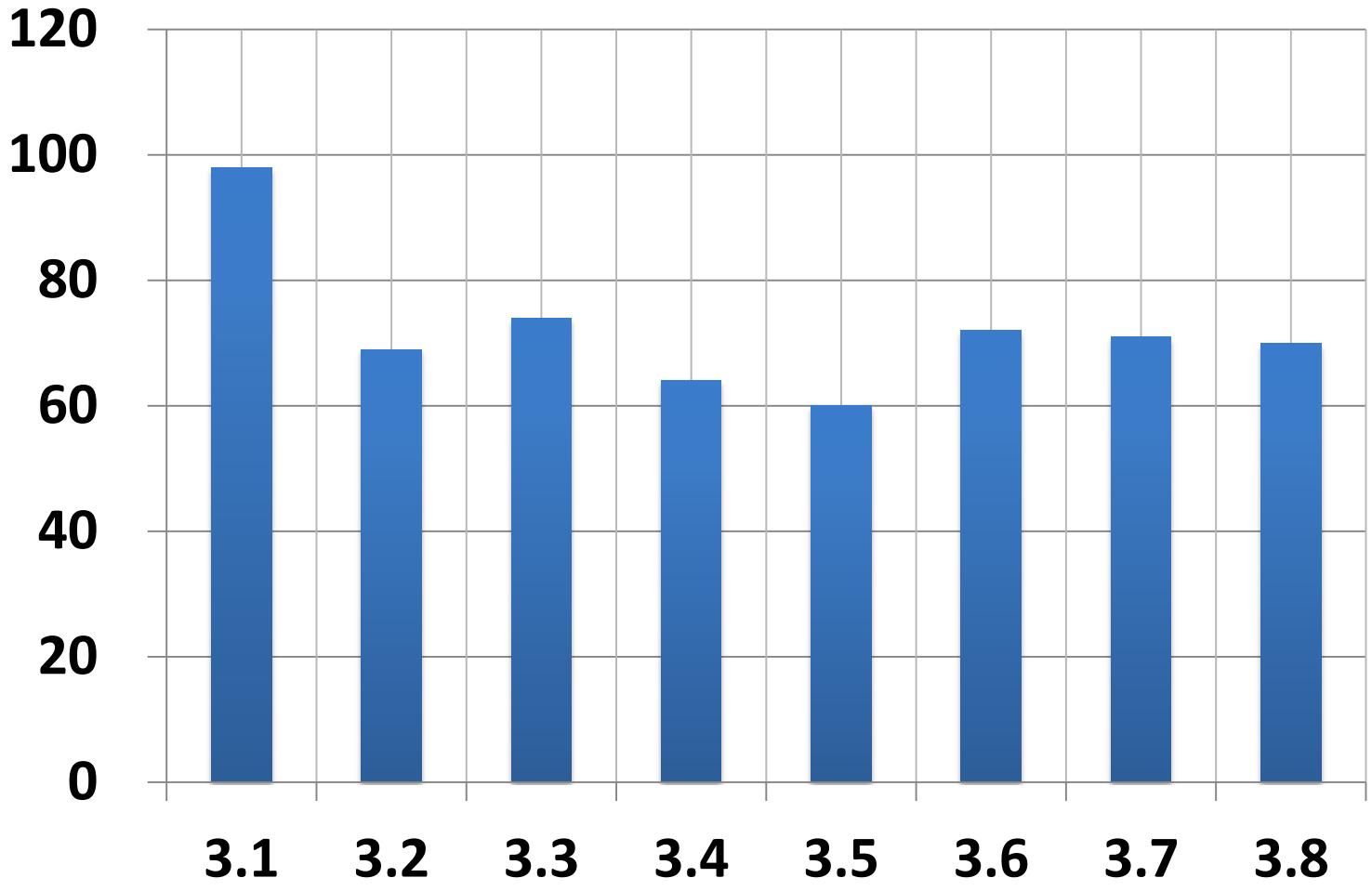
Linux kernel: release date, # of files and lines

Version	Release date	# of Files	# of Lines	
3.8	2013-2-19	41,520	16,416,967	(+1.4%)
3.7	2012-12-11	40,905	16,191,784	(+1.4%)
3.6	2012-10-1	39,733	15,868,122	(+1.7%)
3.5	2012-7-21	39,096	15,596,464	(+1.4%)
3.4	2012-5-22	38,566	15,383,946	(+1.4%)
3.3	2012-3-19	38,082	15,166,160	(+1.1%)
3.2	2012-1-5	37,617	14,998,737	(+1.5%)
3.1	2011-10-28	37,085	14,770,555	

Released
60-70 days

500 Files added,
200,000 Lines of code added
for a version

Linux kernel: Date for release



Stability

- Older kernel is not always stable
- Latest kernel is not always stable
  - Community development is always “Release early, Release often”
  - Many developers are reviewing and changing the code
  - Solving the problem and developing features are always for the latest version of mainline

Stability

- Older kernel is not always stable
- Latest kernel is not always stable
  - Community development is always “Release early, Release often”
  - Many developers are reviewing and changing the code
  - Solving the problem and developing features are always for the latest version of mainline

For the products, we need to back port latest feature to the production kernel
We need to watch the development community to understand what is happening

Maintenance – bug and security fixes

- Number of bugs will be found after the kernel release
- Number of security problem will also be found after the kernel release
- All such problems are fixed in the latest kernel version

Maintenance – bug and security fixes

- Number of bugs will be found after the kernel release
- Number of security problem will also be found after the kernel release
- All such problems are fixed in the latest kernel version

We should watch the community and every bugs and security fixes should be back port to the production kernel. If you will lost the back port, your product includes security hole and bugs. That will be a company's risk

Maintenance – in-house patch

- Everyone have own changes as in-house code
  - Sometime fixes the bugs and add nice features
  - But most of such changes are stay in-house
- Kernel development is moving very fast and changing its code
- So, in case of new product development, in-house code may not be able to apply to the newer target kernel

Maintenance - in-house patch

- In-house patch porting process

While every single in-hose patches
  if a patch cannot apply to the target kernel
    Investigate the reason
      rewrite the patch for target kernel
      test the patch on the target kernel

Maintenance - in-house patch

- In-house patch porting process

Reason may depends on for both in-house patch and kernel itself

While every single in patches
  if a patch cannot pply to the target kernel
    Investigate the reason
      rewrite the patch for target kernel
      test the patch on the target kernel

Maintenance - in-house patch

- In-house patch porting process

Reason m
both in-ho
kernel itse

Engineer who wrote the patch may not in the team

While every single in pat
if a patch cannot pply to th target kernel
Investigate the reason
rewrite the patch for target kernel
test the patch on the target kernel

Maintenance - in-house patch

- In-house patch porting process

Reason m
both in-ho
kernel itse

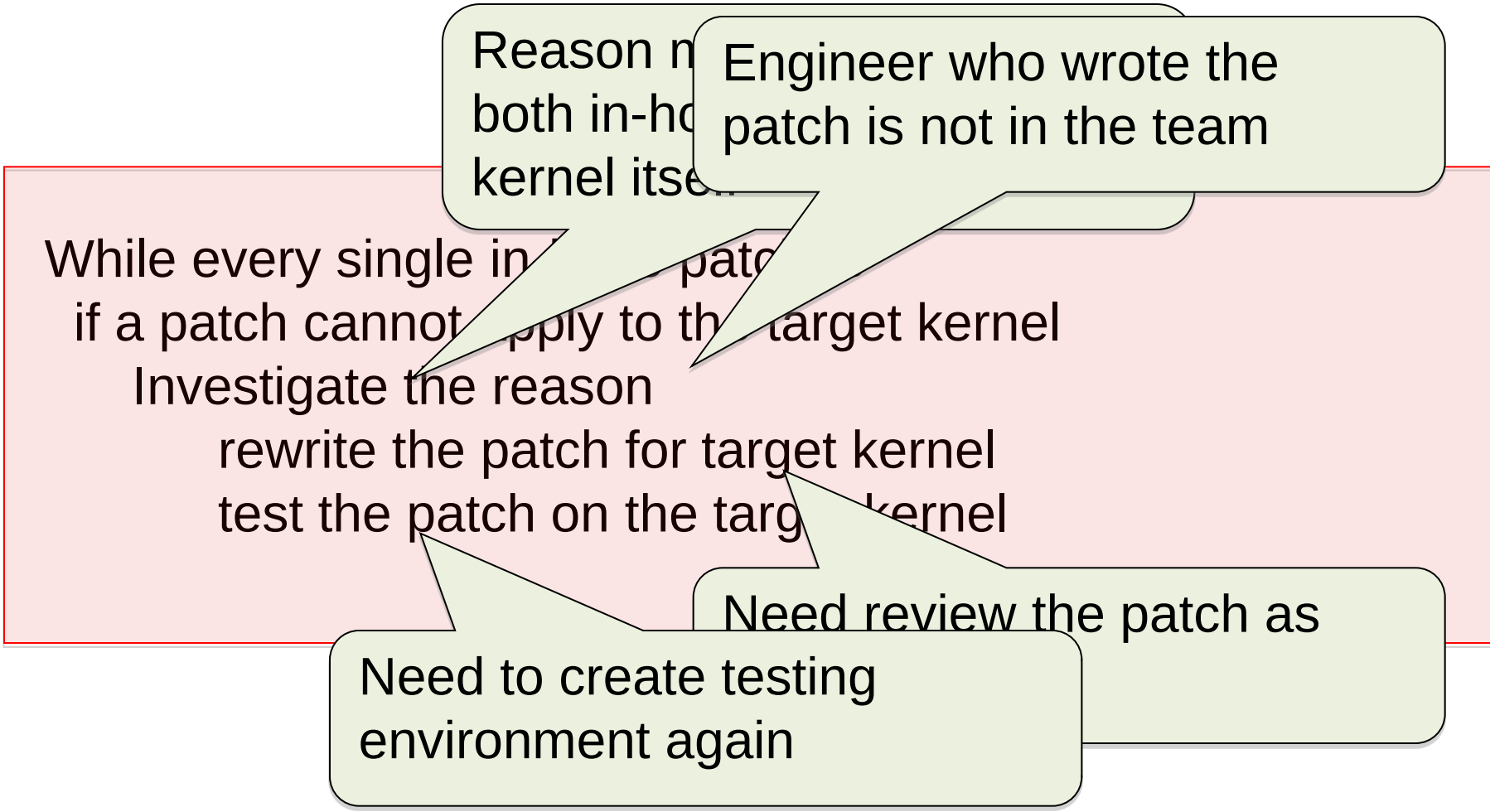
Engineer who wrote the patch is not in the team

While every single in pat
if a patch cannot pply to th target kernel
Investigate the reason
rewrite the patch for target kernel
test the patch on the targ kernel

Need review the patch as the correctness

Maintenance - in-house patch

- In-house patch porting process



Maintenance - in-house patch

- In-house patch porting process

Patch porting works continue for # of patches

rote the

patch is not in the team

kernel itself

While every single in-hose patches

if a patch cannot apply to the target kernel

Investigate the reason

rewrite the patch for target kernel

test the patch on the target kernel

Need review the patch as

Need to create testing environment again

These patch porting work continue for future products whenever in-house code exists

Cost

- Development cost
- Maintenance cost
- HW/Product cost
- Sales/Marketing cost
- …

Cost

- Development cost
  - Specific application or middle ware
  - Tuning for overall system
  - Specific driver for kernel
  - Patch porting to newer kernel
- Maintenance cost
  - Back porting bugs and security fixes
  - Fixes for own application and middle ware
- HW/Product cost
- Sales/Marketing cost

Summary

- Product development is deeply depending on how engineers are participating the to the community
  - Watch the development status
  - How the bugs and security problems are fixed
  - In-house code need to be merged into upstream
  - Decrease the development cost
  - Share information among the engineers will also decrease the development cost

LTSI: Key activities

- Provide a industry managed kernel and maintain Long term stably
- Provide a common place for embedded industry
- Provide place to support upstream activity

LTSI: Industry managed kernel

- LTSI defines common kernel every year and maintains for 2 years
- LTSI adopts community Long term kernel as a base. Therefore, bug/security fixes from upstream are automatically applied to LTSI.
- LTSI kernel = long term + additional patches
- After 2 years term finished, possible to take over maintenance for longer term

Community's long term kernels and Its consumers

- 2.6.27: SUSE11
- 2.6.32: SUSE11 SP1/RHEL 6/Ubuntu10.04 LTS
- 2.6.34: Wind River Linux
- 2.6.35: Embedded usage, Android (Ginger Bread)
- 3.0: ***LTSI***, Android (Ice Cream Sandwich), SUSE11 SP2
- 3.2: Debian7, Bunt 12.04 LTS
- 3.4: ***LTSI,*** Android (Jelly bean)

LTSI: Common place

- LTSI provides the place to share information and experience among industry
  - Mailing list to share problem and discuss how to solve
  - Open Workshop to share status among the industry
  - Closed meeting :ICM (Industry Contact Meeting) for more deeper F2F discussion
- Share the information will reduce the development cost

Open Workshop

- February 21, 2013 3:00pm - 5:00pm

  at Hearst Room, 4th Floor, Park 55 Hotel
- The workshop will cover :
  - Brief Updates of LTSI
  - Updates from a partner project: Doctor
  - Discussion on after release patch acceptance policy
  - Discussion on Super Long Term Support (over 2 years support)
  - Discussion on the next LTSI release

LTSI: Support upstream activity

- Developer in the embedded industry need to know how to work with community
- LTSI provides help upstream activities
  - Provide suggestion how their patches can be merged into upstream
  - Review and discussion for proposed patches to be merged
  - Many of discussion under way
- Merging patches to upstream is also reduce the cost of each companies

LTSI Use Case Program

- LTSI would like to expand use case
- We would ask you to port LTSI to your preferred OS?
  - Android, CyanogenMOD, FirefoxOS
  - Gentoo, OpenWRT, XBMC,GeeXBoX
  - Debian, Ubuntu, Fedora, OpenSUSE for ARM
- We will help your activities with providing Board HW and that will be used by your self after the porting finished
- You need to write your poring report to eLinux Wiki
- You need to send patch to LTSI mailing list if you have changed
- Details can be discussed at LTSI Booth 6pm today

How you can participate LTSI

- Follow on Twitter account:

  @LinuxLTSI

- Web:

  http://ltsi.linuxfoundation.org

- Mailing list:

  https://lists.linuxfoundation.org/mailman/listinfo/ltsi-dev

- Git tree :

  http://git.linuxfoundation.org/?p=ltsi-ernel.git;a=summary

THANK YOU