

开源开发中的沟通-- communication

周明辉
北京大学

引言

- 人类的沟通in general
- 软件开发中的沟通
- 开源项目中提交patch时怎么沟通

人类的沟通

人类沟通的复杂性

- 2个人沟通mismatch的例子
- 3个人沟通mismatch的例子
- n个人沟通mismatch的例子
- 为什么? 各自有不同的assumption和mindset.....

What is communication

Defining Communication



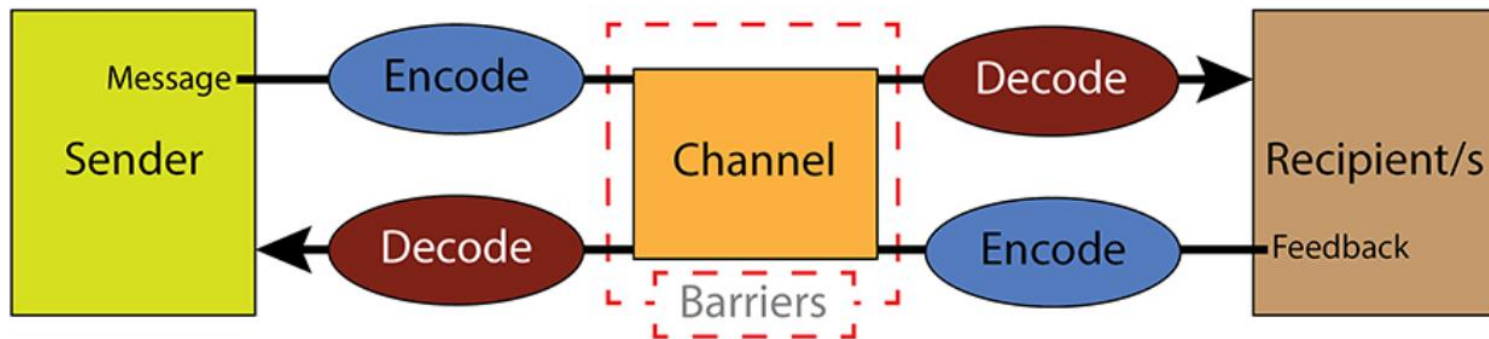
communication, *n.* The imparting or exchanging of information by speaking, writing, or using some other medium. ...The successful conveying or sharing of ideas and feelings.

Oxford English Dictionary



The desired outcome or goal of any communication process is mutual understanding.

The Communication Process



Being able to communicate effectively is the most important of all life skills.

<https://www.skillsyouneed.com/ips/what-is-communication.html>

Diversity about communication definition (wikipedia)

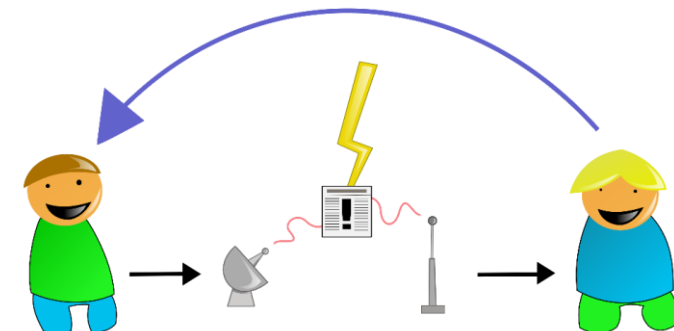
- There are many disagreements about its precise [definition](#).^{[4][5]} John Peters argues that the difficulty of defining communication emerges from the fact that communication is both a [universal](#) phenomenon (because everyone communicates) and a [specific discipline](#) of institutional academic study.^[6]
- Some theorists give very broad definitions of communication that encompass unconscious and non-human behavior.^[17] In this regard, many animals communicate within their own species and even plants like flowers may be said to communicate by attracting bees.^[14]
- Other researchers restrict communication to [conscious](#) interactions among human beings.^{[17][14]} Various characterizations see the communicator's [intent](#) to send a message as a central component. On this view, the transmission of information is not sufficient for communication if it happens unintentionally.^[17]
- The paradigmatic form of communication happens between two or several individuals. However, it can also take place on a larger level, for example, between organizations, social classes, or nations.^[14]
- [Niklas Luhmann](#) rejects the view that communication is, on its most fundamental level, an interaction between two distinct parties. Instead, he holds that "only communication can communicate" and tries to provide a conceptualization in terms of [autopoietic systems](#) without any reference to consciousness or life.^[21]
- John Peters sees communication as "an apparent answer to the painful divisions between [self](#) and other, private and public, and inner thought and outer world."

googleTranslation:

- 关于其精确定义存在许多分歧。约翰·彼得斯认为，定义沟通的困难在于沟通既是一种普遍现象（因为每个人都在交流），也是一门学术研究的特定学科。
- 一些理论家给出了非常广泛的沟通定义，包括无意识和非人类行为。在这方面，许多动物在它们自己的物种内交流，甚至可以说像花这样的植物通过吸引蜜蜂来交流。
- 其他研究人员将交流限制在人类之间有意识的互动中。各种特征将传播者发送消息的意图视为中心组件。根据这种观点，如果无意中发生的信息传输不足以进行通信。
- 典型的交流形式发生在两个或几个人之间。然而，它也可以发生在更大的层面上，例如在组织、社会阶层或国家之间。
- **Niklas Luhmann** 拒绝认为沟通在最基本的层面上是两个不同方之间的互动。相反，他认为“只有交流才能交流”，并试图根据自创生系统提供概念化，而不涉及任何意识或生命。
- 约翰·彼得斯（**John Peters**）将沟通视为“对自我与他人、私人与公共、内心思想与外部世界之间痛苦分歧的明显答案”。

Shannon and Weaver Model (Wikipedia, accessed 2022/10.25)

- The first major model for communication was introduced by [Claude Shannon](#) and [Warren Weaver](#) for Bell Labs in 1949^[41] The original model was designed to mirror the functioning of radio and telephone technologies. Their initial model consisted of three primary parts: sender, channel, and receiver. The sender was the part of a telephone a person spoke into, the channel was the telephone itself, and the receiver was the part of the phone where one could hear the other person. Shannon and Weaver also recognized that often there is static that interferes with one listening to a telephone conversation, which they deemed noise.
- In a simple model, often referred to as the transmission model or standard view of communication, information or content (e.g. a message in [natural language](#)) is sent in some form (as [spoken language](#)) from an emitter/sender/encoder to a destination/receiver/decoder. This common conception of communication simply views communication as a means of sending and receiving information. The strengths of this model are simplicity, generality, and quantifiability. Claude Shannon and Warren Weaver structured this model based on the following elements. (These elements are now understood to be substantially overlapping and recursive activities rather than steps in a sequence.)
 1. [The formation of communicative motivation or reason](#).
 2. [Message](#) composition (further [internal](#) or [technical](#) elaboration on what exactly to express).
 3. Message encoding (for example, into [digital data](#), [written text](#), [speech](#), [pictures](#), [gestures](#) and so on).
 4. [Transmission](#) of the encoded message as a sequence of signals using a specific [channel](#) or [medium](#).
 5. [Noise sources](#) such as natural forces and in some cases human activity (both intentional and accidental) begin influencing the quality of signals propagating from the sender to one or more receivers.
 6. [Reception](#) of signals and reassembling of the encoded message from a sequence of received signals.
 7. Decoding of the reassembled encoded message.
 8. [Interpretation](#) and [making sense](#) of the [presumed](#) original message.
- Shannon and Weaver argued that there were three levels of problems for communication within this theory.^[citation needed]
 - The technical problem: how accurately can the message be transmitted?
 - The semantic problem: how precisely is the meaning conveyed?
 - The effectiveness problem: how effectively does the received meaning affect behavior?



41. Shannon, C.E., & Weaver, W. (1949). *The mathematical theory of communication*. Urbana, Illinois: University of Illinois Press

42. Reddy, Michael J. (1979). "The Conduit Metaphor -- A Case of Frame Conflict in our Language about Language." In *Metaphor and Thought*, Andrew Ortony, ed. Cambridge UP: 284-324.

人类沟通的案例

- 一个同学说的北大和隔壁的招生
 - Method 1: 讲科研, 高大上, 听不懂
 - Method 2: github账号, 开源项目, 本土操作系统, ...
- 我跟国际同事的gossip communication:
 - 我跟一位立陶宛同事说: 你们国家的语言是俄语吗?
 - 我跟一位美国同事说: 哦, 你是说那位old lady, 我知道她...
 - 对方大惊失色! If i don't know you better, I'll be angry.
- 沟通有障碍, 因为文化、语言、背景、...
- 我们为什么而沟通? 沟通目的也会决定沟通质量。

沟通中的黄金定律：用别人喜欢被对待的方式来对待他们



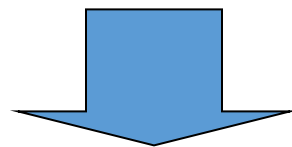
一只载着各国乘客的豪华轮船撞上了冰山，马上就要沉了。船长要鼓励乘客跳海逃生，该如何说？

（有目的的沟通）

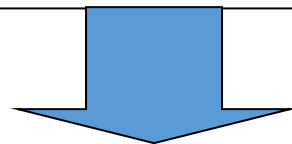
- 对中国人说：“看起来很好吃的鱼在游泳呢”
- 对英国人说：“作为绅士这个时候可是要跳下去的哦”
- 对德国人说：“按照规则是应该跳的”
- 对意大利人说：“刚才一个美女跳下去了”
- 对美国人说：“想当英雄吗，那就跳下去吧”
- 对俄罗斯人说：“伏特加的瓶子被冲走了，现在追还来得及”
- 对法国人说：“请千万不要跳下去”
- 对日本人说：“大家都跳了，你还不跳？”

软件开发中的沟通

软件一个显著特点是**复杂性**，而且**程序复杂性**将随着程序规模的增加而呈**指数上升**。



为了在预定时间内开发出规模庞大的软件，必须由**许多人分工合作**。

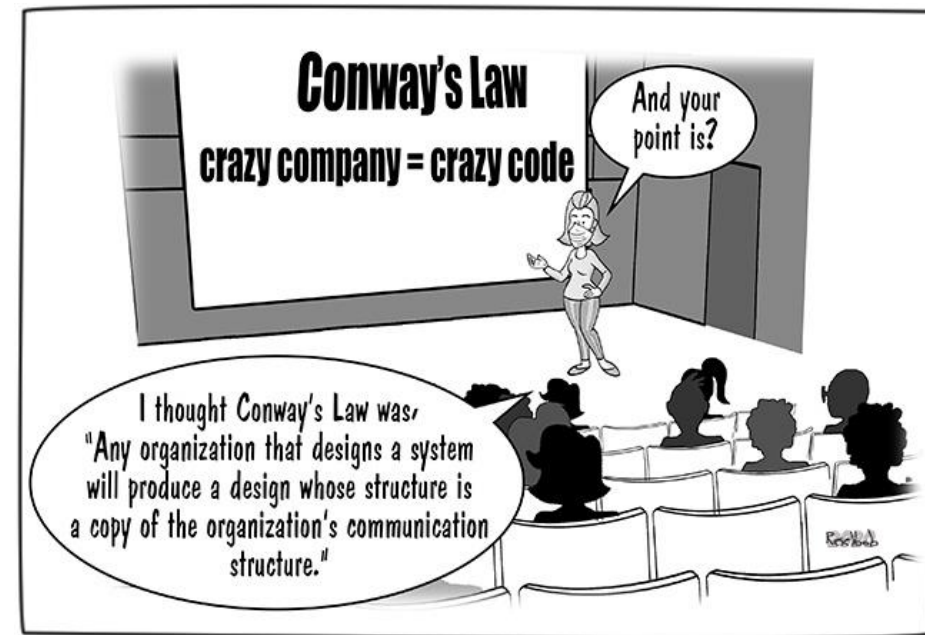


如何保证个体完成的工作合在一起确实能构成一个**高质量的大型软件系统**，是一个**极端复杂困难**的问题，不仅涉及许多**技术问题**，还涉及**管理协作问题**，并且两者相辅相成：Conway law。

conway's law

"How Do Committees Invent?" Melvin Conway, 1968

<http://tinyurl.com/conwayslaw>



PARAPHRASED

CONWAY'S LAW

THE STRUCTURE OF SOFTWARE WILL MIRROR THE STRUCTURE OF THE ORGANISATION THAT BUILT IT *for example*

ORGANISATION

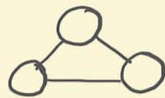


SMALL DISTRIBUTED TEAMS

are more likely to produce



SOFTWARE



MODULAR, SERVICE ARCHITECTURE



LARGE COLOCATED TEAMS



MONOLITHIC ARCHITECTURE

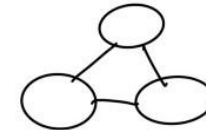
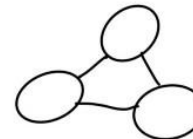
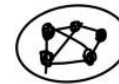
sketchplanations

CONWAY'S LAW

THE ORGANISATION DESIGN

is likely to be mirrored in

THE PRODUCT DESIGN



管理复杂度的技术手段

- 管理复杂度是软件开发的首要技术目标。
- 怎样可以控制复杂度？经典原则：
 - 模块化
 - 抽象：抽象通过提供一个可让你忽略实现细节的模型来管理复杂度，而封装则强制阻止你看到细节。P.J Plauger: 如果必须要看到底层实现才能理解所发生的事情，那还算不上抽象。
 - 逐步求精
 - 信息隐藏和局部化
 - 模块独立
- 上述技术手段足够吗？

问题：如何组建一支高效的团队？采用什么管理模式？

[team]
一个人的能力再强也
无法顾及所有的事情，
必须得有一个很强的团队



团队

软件团队的模式

- **主治医师模式 (Chief Programmer Team)**

- 首席程序员负责主要模块的设计和编码，其他成员从各种角度支持他的工作 (后备程序员、系统管理员、工具开发、编程语言专家、业务专家)
- 例子：
 - Frederic Brooks Jr., IBM System 360的开发

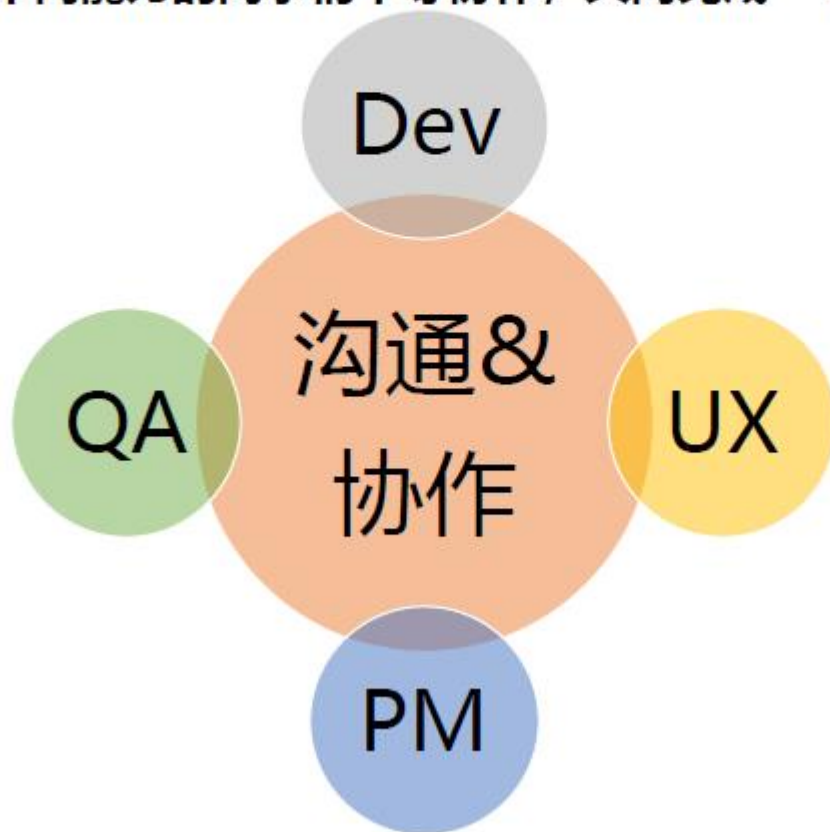
- **明星模式 (Super-star Model)**

- 主治医师模式运用到极点，就蜕变为明星模式
- 例子：
 - 刘翔——翔之队

软件团队的模式

- 功能团队模式 (Feature Team)

- 具备不同能力的同事们平等协作，共同完成一个功能



软件团队的模式

- 官僚模式 (Bureaucratic Model)

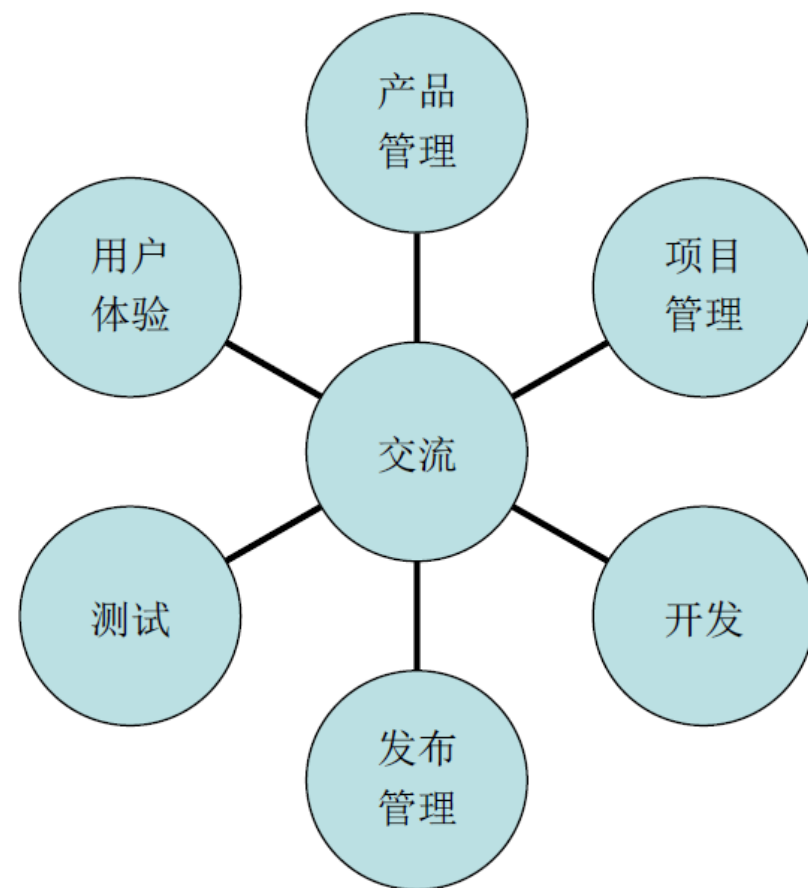
- 类似于大机构的组织架构，层层报告
- 成员之间不仅有技术上的合作和领导关系，还有组织上的领导关系
- 跨组织合作比较困难

你会选择什么样的团队结构？

- 取决于上下文
- 沟通是中枢

非常抽象的解决复杂度的方案！

- 开源项目中人们会避免沟通，但面向任务仍然需要沟通，什么是其必要沟通内容？



敏捷 (Agile) 开发流程

- “敏捷开发流程” 是一系列价值观和方法论的集合
 - 始于2001年

现有的做法	敏捷的做法
流程和工具	个人和交流
完备的文档	可用的软件
为合同谈判	与客户交流
执行原定计划	响应变化

敏捷开发原则

- 业务人员和开发人员在项目开发过程中应该每天共同工作
- 以有进取心的人为项目核心，充分支持信任他们
- 无论团队内外，面对面的交流始终是最有效的沟通方式

沟通是核心要素；
但成功取决于执行者。
仍然是抽象的解决复杂度的方案。

为了达成有效沟通的技术手段

- 版本管理
- 任务追踪
- CI/CD
- 其他???

什么是好的沟通？

Linux kernel社区中提交patch时的沟通best practice

Tan and Zhou. How to Communicate when Submitting Patches: an Empirical Study of the Linux Kernel. ACM on Human-Computer Interaction. CSCW 2019

Text is the Only Means to Communicate when Submitting Patches (CSP)



Submitting Patches

arm/i915/execlists: Simply walk back along request timeline on reset

Chris Wilson

The request's timeline will only contain requests from this context, in order of execution. Therefore, we can simply look back along this timeline to find the currently executing request.

If we do find that the current context has completed its last request, that does not imply that all requests are completed in the context, so only advance the ring->head up to the end of the known completions!

Signed-off-by: Chris Wilson <chris@chris-wilson.co.uk>

drivers/gpu/drm/i915/gt/intel_lrc.c | 29 ++++++
1 file changed, 12 insertions(+), 17 deletions(-)

```
diff --git a/drivers/gpu/drm/i915/gt/intel_lrc.c b/drivers/gpu/drm/i915/gt/intel_lrc.c
index 16340740139d..60472b83afaf 100644
--- a/drivers/gpu/drm/i915/gt/intel_lrc.c
+++ b/drivers/gpu/drm/i915/gt/intel_lrc.c
@@ -252,22 +252,15 @@ static void mark_eio(struct i915_request *rq)
{
    static struct i915_request *active_request(struct i915_request *rq)
    {
        const struct intel_context * const ce = rq->hw_context;
        struct i915_request *active = NULL;
        struct i915_request *active = rq;
        struct list_head *list;

        if (!i915_request_is_active(rq)) /* unwound, but incomplete! */
            return rq;

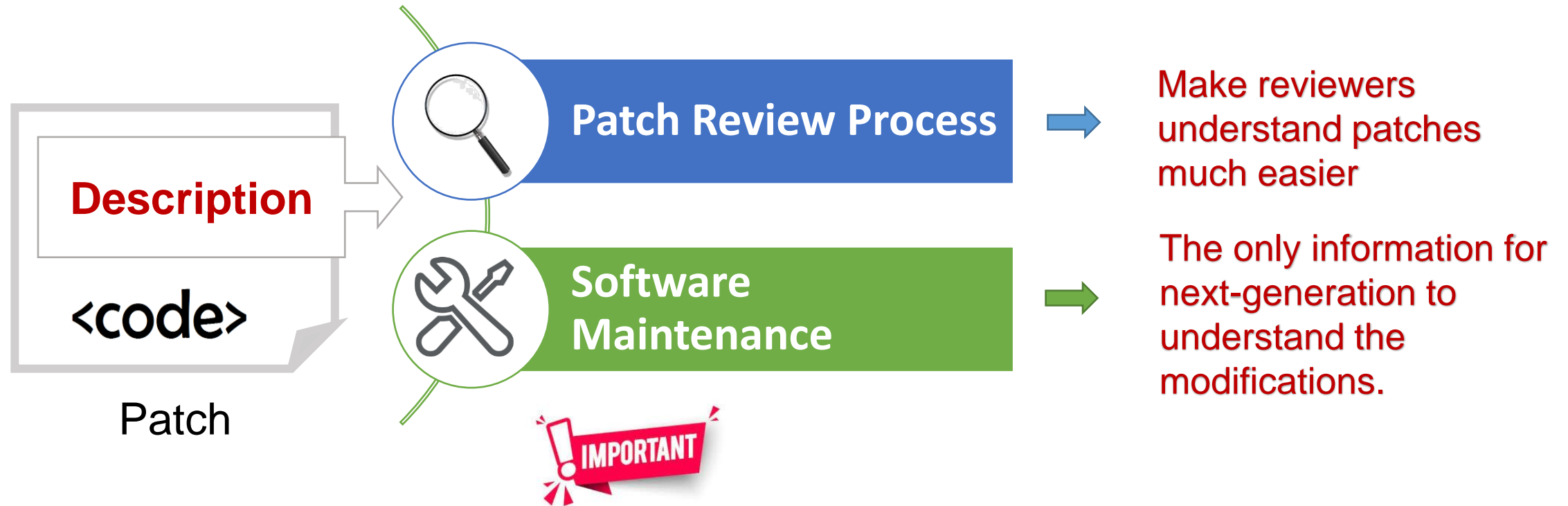
        rcu_read_lock();
        list = &rcu_dereference(rq->timeline)->requests;
        list_for_each_entry_from_reverse(rq, list, link) {
```

Description

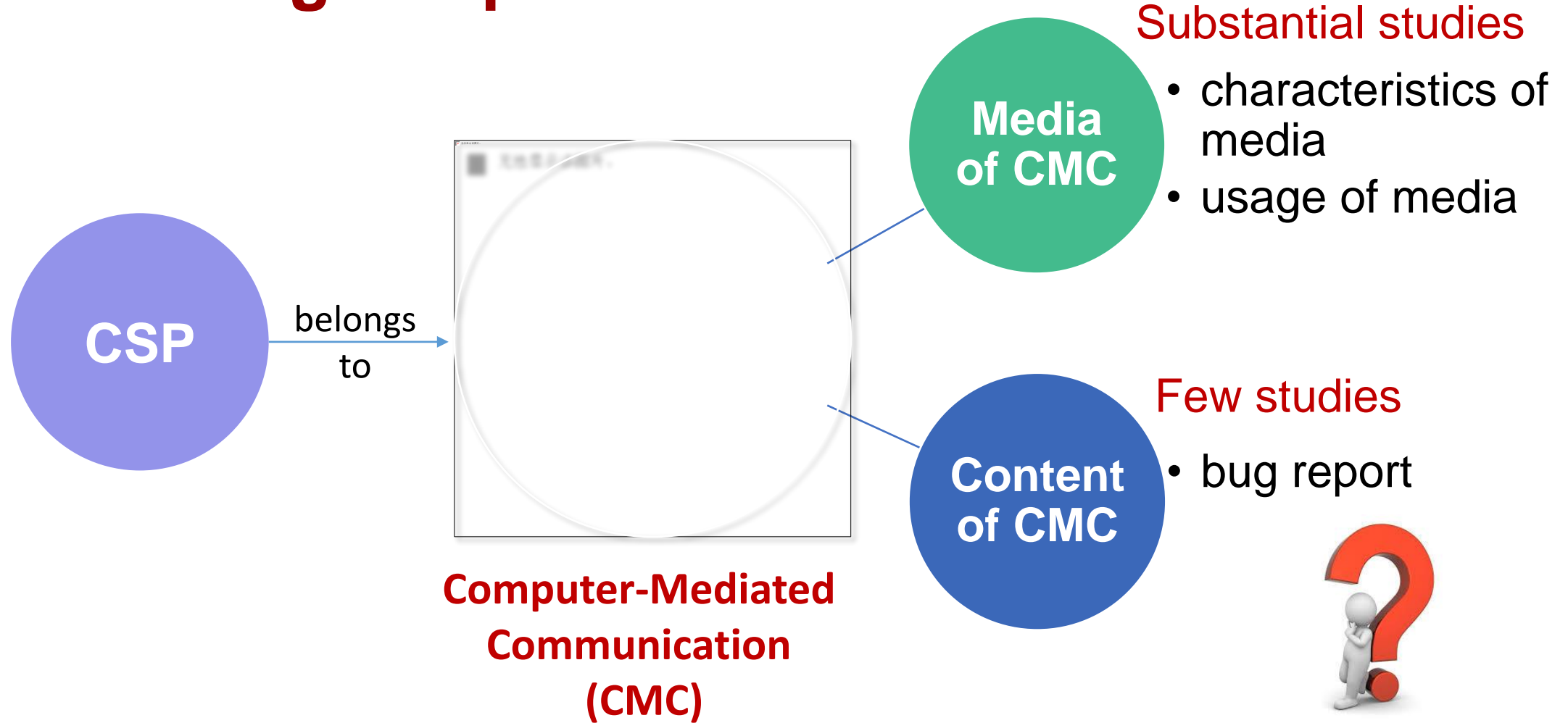
Code

A patch from the Linux kernel

Communication When Submitting Patches (CSP) is very Important

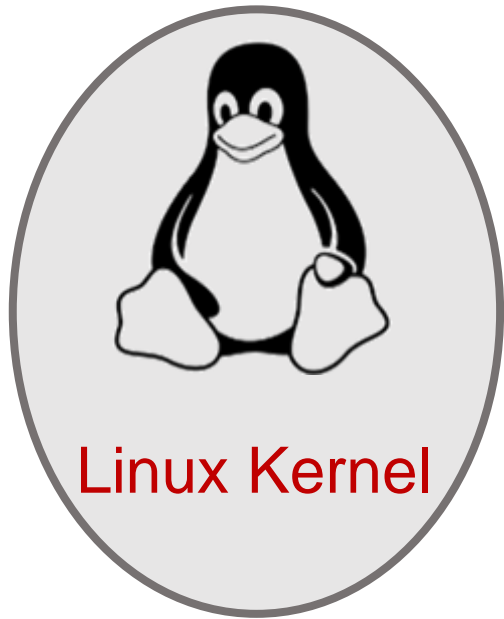


Knowledge Gap



Study Design

Dataset



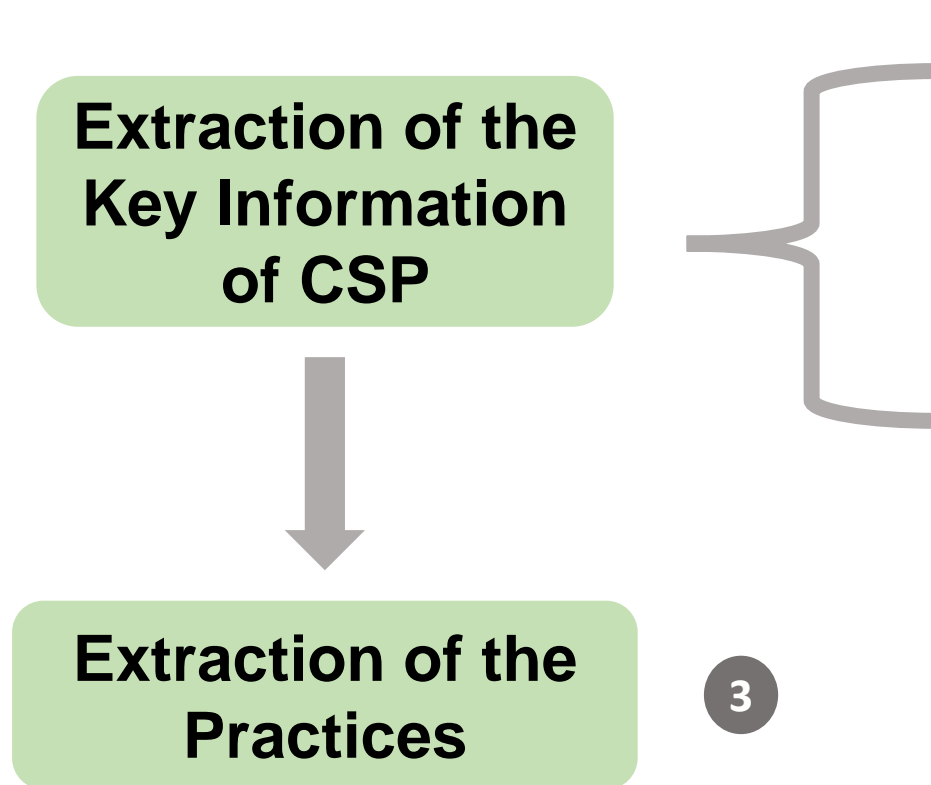
23 online documentations

(rules or official instructions of patch submission)

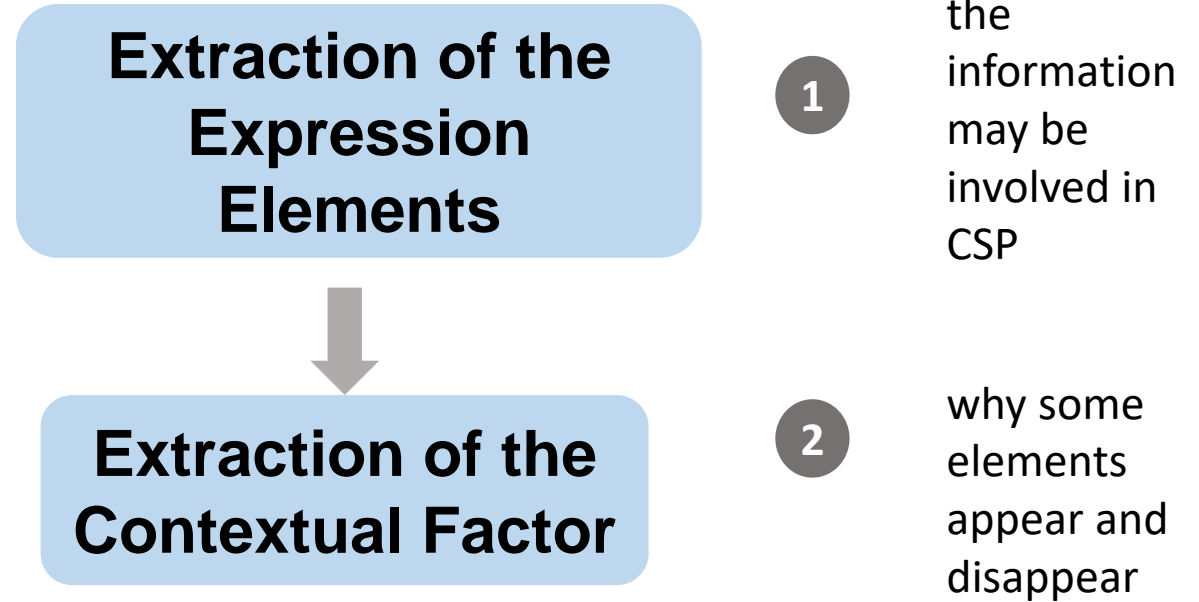
500 accepted patches [Xu and Zhou MSR'18]

200 patches rejected because of communication
(focus on the patches that have multiple versions but only differ in their description)

Study Design

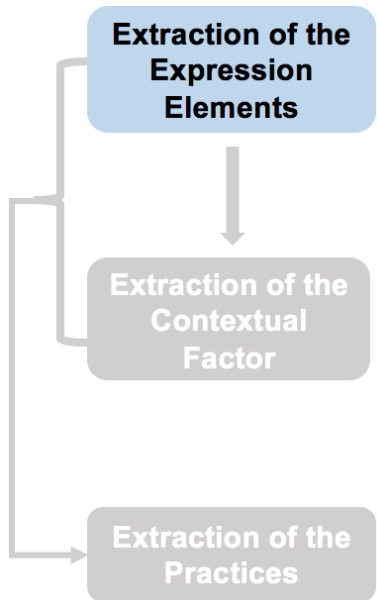


Thematic analysis



Extraction of the Expression Elements

Extraction resources: *online documents and 500 accepted patches*



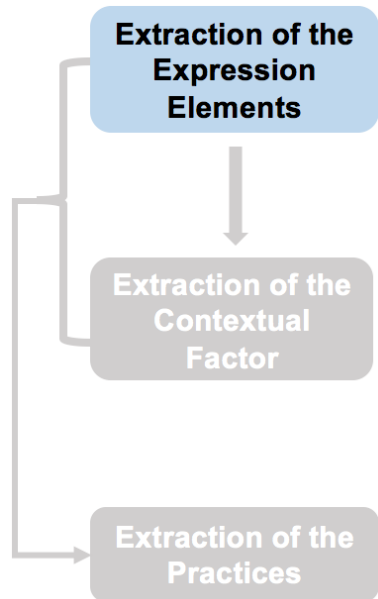
- **(Patch #747511)** Recent fixes for iATU unroll support introduced a bug that causes asynchronous external abort in Keystone PCIe h/w which doesn't have ATU port and the corresponding register.

motivation

Seriousness
of bug

Expression Elements

17 elements falling into 4 major themes



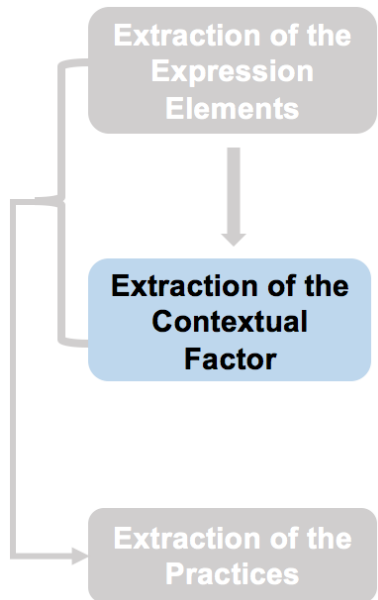
Themes		Elements
Timing of CSP		Timing of submission
Granularity of CSP		Separation of change
Content of CSP	Description of motivation	Motivation
	Description of problem	Bug reproduction; Seriousness of bug; Related commit
	Description of implementation	Technical details; Brief introduction of functions; Improvement; Trade-off; Reference
	Description of quality	Superiority; Test/review; Limitation
Manner of CSP	Language	Mood of speaking
	Satisfy basic requirement of community	Basic requirement of subject; Width of description

Extraction of the Contextual Factors



Extraction of the Contextual Factors

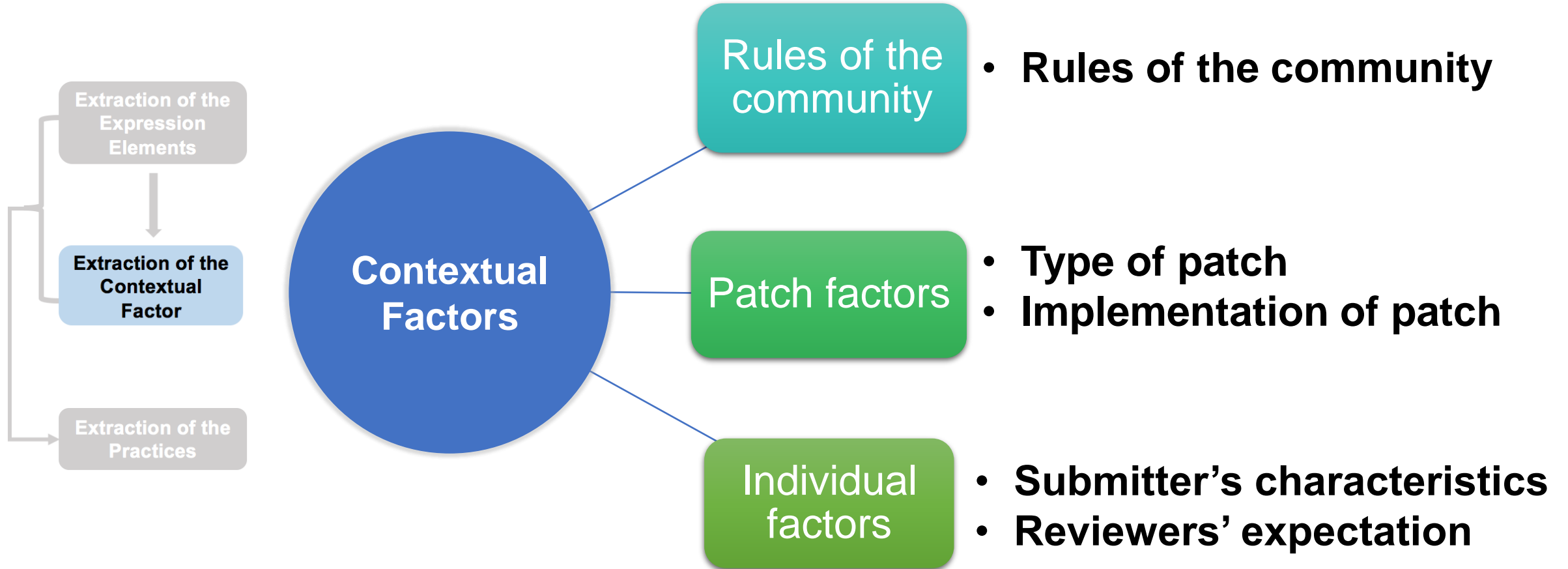
Extraction resources: *500 accepted patches in the last step and their expression elements*



Reasons that are Associated with the Occurrence of the Expression Elements

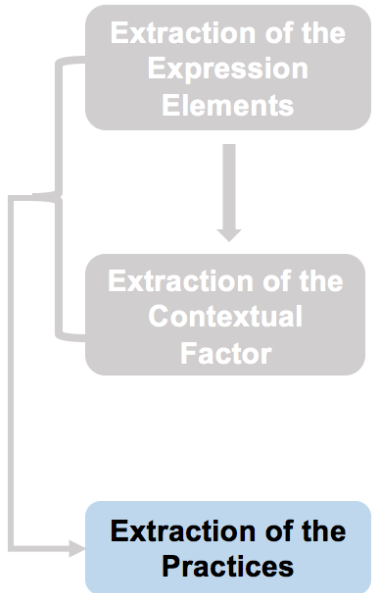
Elements	Reasons for CSP containing this element	Reasons for CSP not containing this element
Timing of CSP	Community's requirement: Only patches of bug fix can be merged in merged window. Others should be submitted outside the window. (500)	None.
Separation of change	It is the requirement of the community. (484)	The subsidiary modifications are simple, e.g., fix a typo. (16)
...

Contextual Factors



Practices for CSP

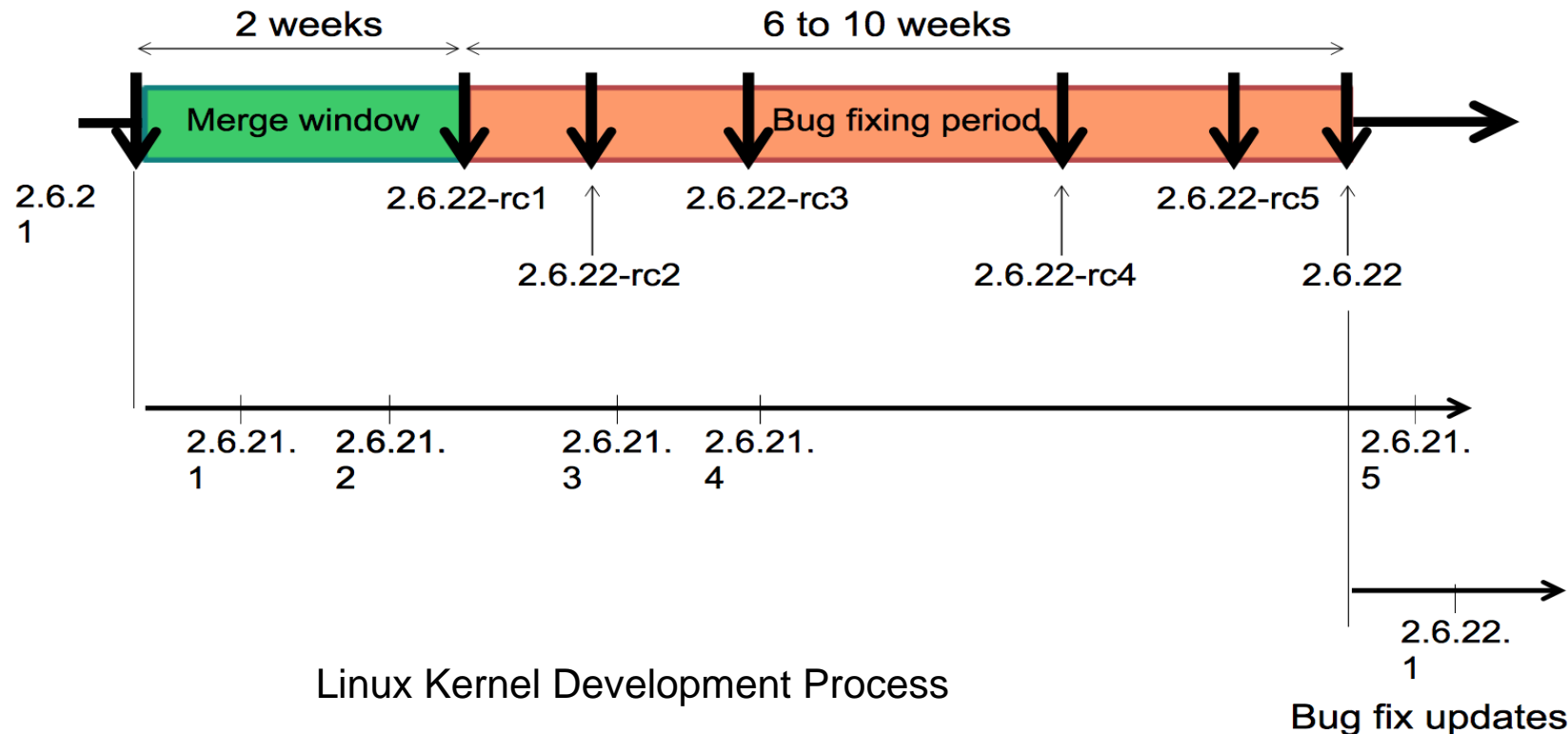
- We summarized **18 preliminary practices**. After performing a survey, we dropped one practice and eventually obtained **17 practices**.



Categories of the Practices	Count
Practices Related to <i>Rules of the Community</i>	4
Practices Related to <i>Implementation of the Patch</i>	8
Practices Related to <i>Type of the Patch</i>	5
Practices Related to <i>Individual Factors</i>	0

Practices Related to Rules of the Community

1 Choose the right timing.



rc: release candidate

Practices Related to Implementation of the Patch

3

If there is a cost after applying the patch, provide the trade-off.

Add single-element dequeue functions to rcu_segcblist. It is **less efficient** than using the extract and insert functions, but allow **more precise** debugging code. (Patch #745004)

This limitation should be explained in the cover letter and changelog.”
(Patch #759033)



Comment for a poor CSP

Practices Related to Implementation of the Patch

4

If the patch refers to other information, e.g., commits, email message, or links to related information, include it.

Commit [e21d2170f36602ae2708](#) (“video: remove unnecessary platform_set_drvdata()”) removed the unnecessary platform_set_drvdata(), but left the variable “dev” unused, delete it. (Patch #2833310)

SHA-1 ID
and one-line
summary

Practices Related to Type of the Patch

5

Provide sufficient reasons why this new feature is required, especially for the complex features.

This adds a device tree definition file for LEGO MINDSTORMS EV3.
(Patch #728065)



Modifications: 457 lines of code



Can you clarify why we need it?



The reason can be sufficient if s/he can give evidence similar to “[w]hat a bad effect there is without it” or “[i]t can support/enable ...”.

Practices Related to Type of the Patch

6

For the patches that are considered trivial or that simplify the code, just indicate what you did.

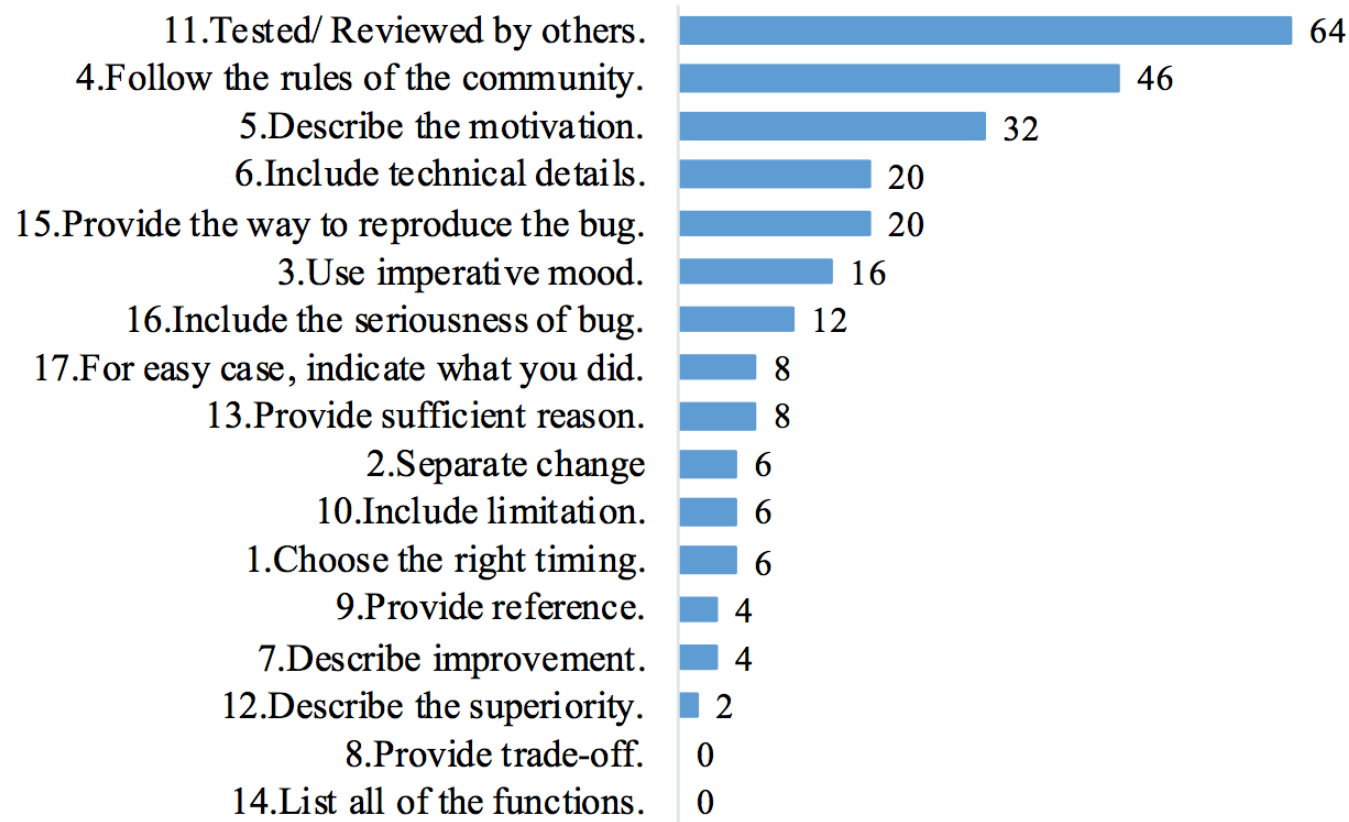


Trivial x to spelling
mistake in dev_dbg
message. (Patch
#10576333)



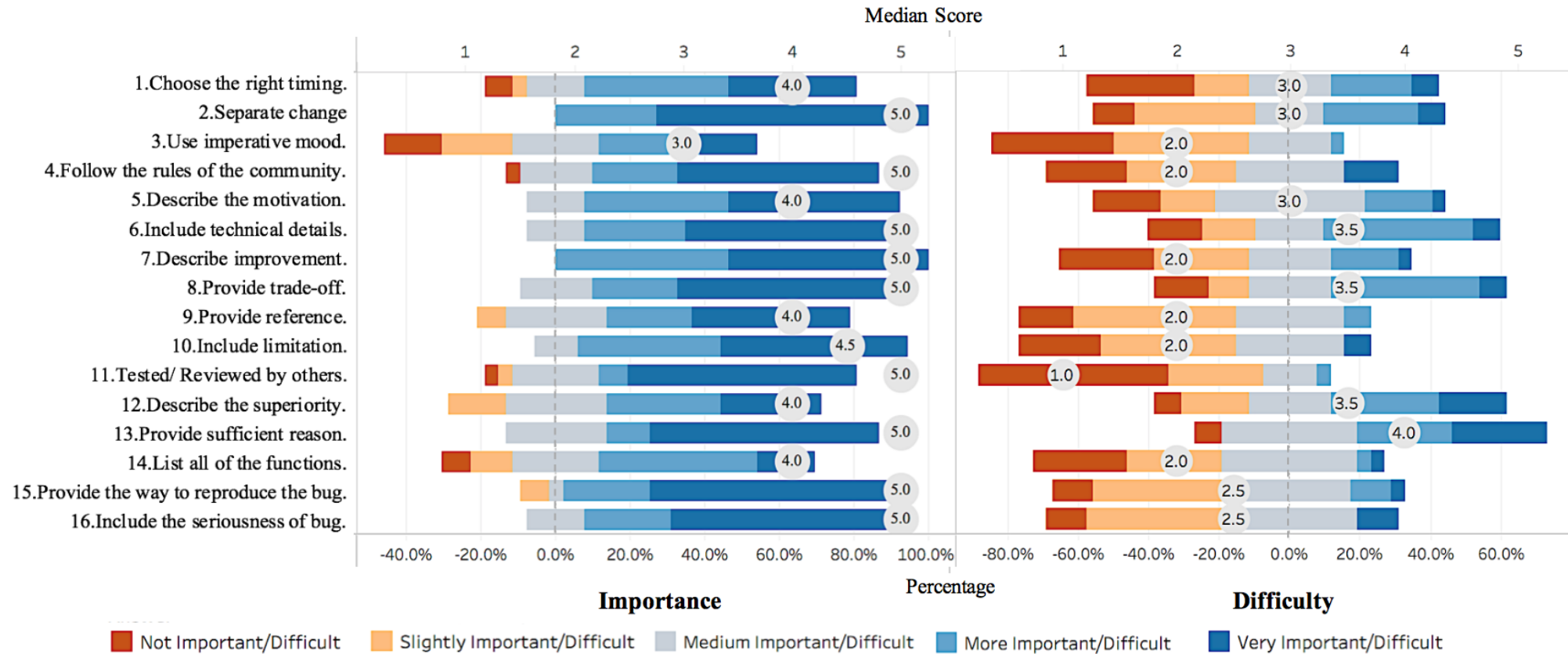
Patches Rejected Because of Poor CSP

200 rejected
patches



Number of the Patches not Following the Specific Practices

Developers' View of These Practices



The practices are effective and almost all the practices are reasonably applicable.

Developers' View of These Practices

Two candidate practices aiming to save time are hard to evaluate.

1. [i]f the feature has been discussed previously, or it's a widely recognized feature, or you (the author of the patch) have commit right, you may omit providing the reason.



meaningless: 88%; not significant: 20%

2. [f]or the patches such as: easy case, and simplify code, just express what you did.



Significant: 88%

Implications



Developers

- 17 practices for CSP



Reviewers

- Facilitating review process
- Potential risk – *Individual Factors*



Tool designers

Commitizen
Zen-like commit messages
for internet citizens

interactively
provide
Contextual Factors to guide
CSP



open source

Open source
communities

- **For the Linux kernel:** five new valuable practices ; potential risks
- **For others:** practices related to *Content of CSP* (13 practices)

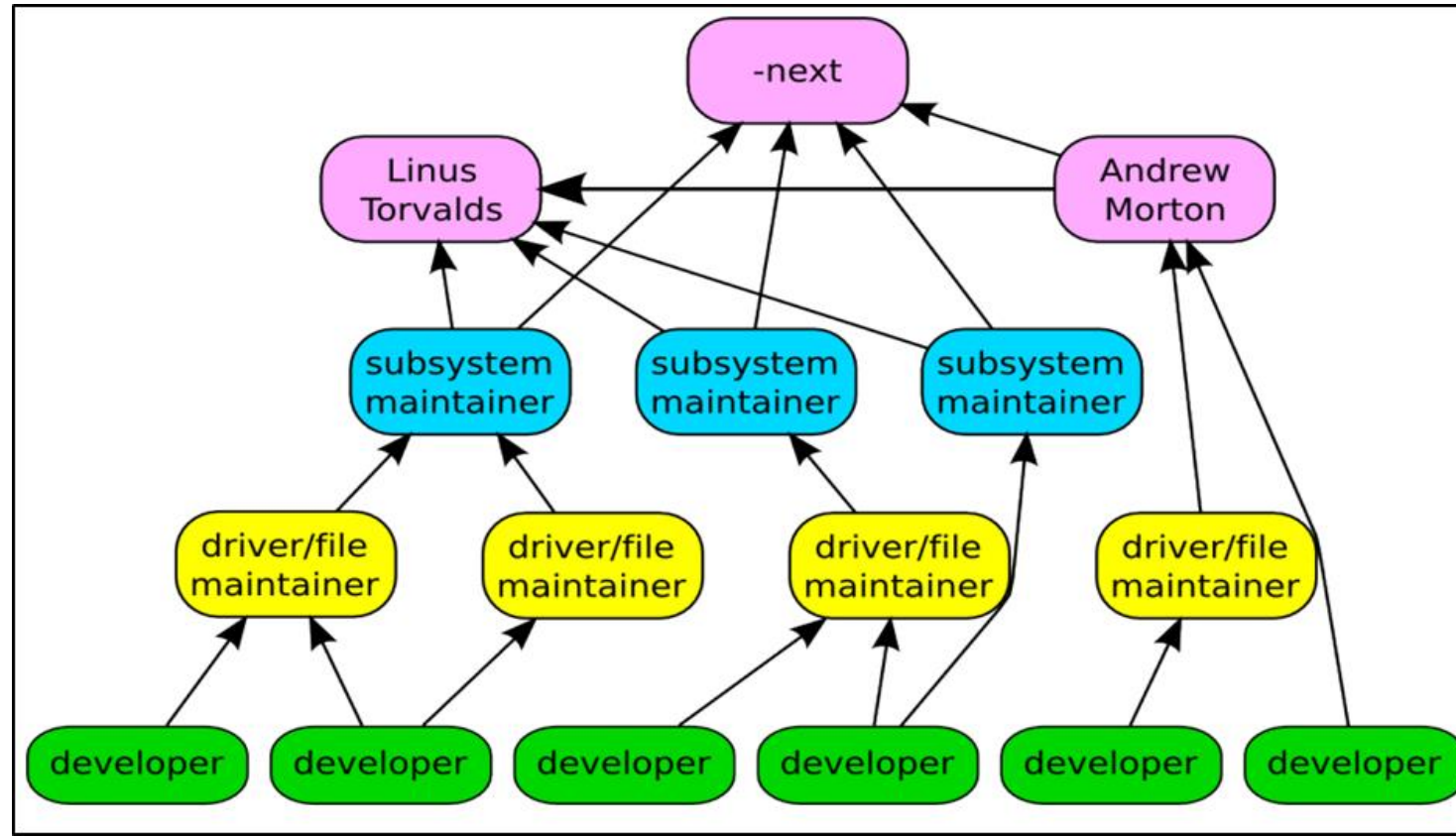


CSCW
researchers

- Bridge the **knowledge gap** of *Content of CMC*
- **Research framework**
(*Expression Elements -> Contextual Factors -> Practices*)

关于沟通的其他

Linux community has been employing hierarchy model for years



开源参与的沟通和协作

- 开源开发：小内核，驱动大外围.....
 - maintainer任务繁重，信息淹没
 - silence protocol
- Keypoint: 无论个体多么英明神武，其负载度是有限的

On Linux kernel maintainer scalability



By **Jonathan Corbet**
October 12, 2016

[LinuxCon Europe](#)

LWN's [traditional development statistics article for the 4.6 development cycle](#) ended with a statement that the process was running smoothly and that there were no process scalability issues in sight. Wolfram Sang started his 2016 [LinuxCon Europe](#) talk by taking issue with that claim. He thinks that there are indeed scalability problems in the kernel's development process. A look at his argument is of interest, especially when contrasted with another recent talk on maintainer scalability.

Beyond changesets merged

Sang's core point is that looking at the number of patches merged only tells part of the story; it says nothing about what had to happen to get those patches into the mainline. Looking at the last few years' worth of development cycles, he noted that relatively few patches carry tags beyond the Signed-off-by applied by the developer and the committer. In particular, around the 3.0 days, only about 20% of the patches in the mainline had an Acked-by, Reviewed-by, or Tested-by tag indicating that anybody other than the maintainer had seriously looked at them. That number is closer to 40% in current kernels, he said; it is a clear improvement, but still does not make him happy. For a properly scalable kernel process, he said, we should have much higher levels of review by developers who are not the subsystem maintainer.

Another metric one can look at is the time difference between the date on the patch and the date on which it was first committed to a git tree. The Ethernet driver maintainers, he said, are heroes: 80% of all the patches were accepted within two weeks. A number of other subsystems do not do anywhere near as well, and some have gotten significantly worse. I2C, Sang's own subsystem, has stayed about the same over the last three years, which surprised him. As the workload has increased, it has come to feel like things are getting much worse.



The time-to-commit metric may be useful, but it is not without its flaws. The final version of a patch may have been committed fairly quickly, but previous versions could have languished without review for a long time. Patches that are rejected or that get lost are not considered at all.

One way to try to get a better handle on things is to look at the [Patchwork](#) systems for the subsystems that use it, and, in particular, to look at the backlog of patches found there. For [I2C](#), it shows a relatively low backlog until about 3.16, when he gave up on trying to keep up with the flow and fell behind. The [ACPI subsystem](#) has an amazing backlog of zero. The relevant maintainer (Rafael Wysocki) was in the room; he noted that it depends on how a subsystem uses Patchwork. He said that he quickly marks a lot of patches as inapplicable; Sang replied that he doesn't even have the time to do that. The [ext4 filesystem](#) shows a linear growth in its backlog, up to about 800 patches currently. The numbers for several other subsystems were shown; almost all of them are going up.

沟通中的误区

- 我是用户！他既然是项目的maintainer，他应该搭理我
- 我做了这么多努力！他凭什么不搭理我
- 我的英语很差，我不知道该怎么说！

认知阻力

白纸和笔	早期版本的Word 软件	2007之后的Word 软件	LaTex 编辑器
估计大致位置， 然后开始写	先写文字，然后在工具栏或菜单中选择“居中对齐”的功能；也可以先选择功能，再写文字；有些用户敲好些空格，然后开始打字，这并不是最正确的做法但结果大致可以接受	（左边的方法同样适用，也可以在文档正中双击鼠标，再开始写文字	在合适的文字前后加上 <code>\begin{center}</code> 和 <code>\end{center}</code> 的标签
认知阻力：小， 用户行为和结果都能非常自然地感知	认知阻力：稍大；用户需要了解“居中”是一个格式的操作；先敲很多空格的做法符合用户对电脑操作的认知，在英文打字机上也是如此操作。	认知阻力：小；几乎和白纸和笔的操作一样	认知阻力：大；用户需要理解LaTex 处理文字的各种规定和标签；用户还要了解 - 有些文字不是正文，而是格式的标记。而且结果不是所见即所得的。不经过一段时间的培训和练习，用户不能够完成这一简单的任务。

关于沟通：
若想得到回应，请让对方的认知阻力尽可能小

沟通中的Best practice

- Explain you are new :
 - It's my first time to make contributions to open source...
 - I'm new to the project, I often use the software and I want to contribute something back to the project.
- Do your homework before asking questions:
 - I have done everything I could to understand the problem, including: 1), I read readme, contributing.md, and, 2), I surveyed the Stackoverflow, and even asked chatGPT, however, I still can't figure out what the problem is.
- Show your effort:
 - My English is poor, but I'll try to talk in code:

致谢：
谭鑫