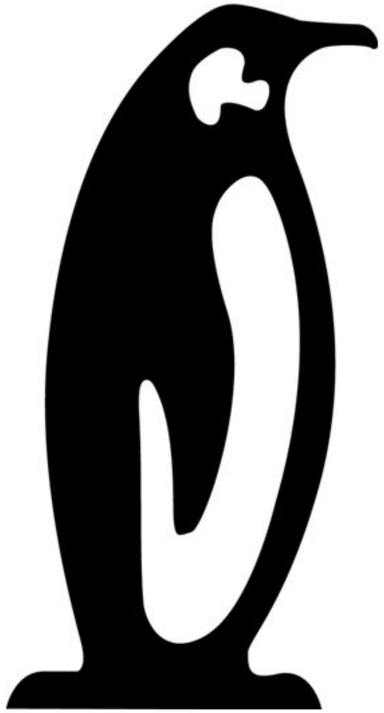

Linux Plumbers Conference

Vienna, Austria | September 18-20, 2024

Improving kernel design documentation and involving experts

Gabriele Paoloni
Senior Principal Software Engineer - Red Hat

LINUX PLUMBERS CONFERENCE | Vienna, Austria
Sept. 18-20, 2024

Who am I ?

Gabriele Paoloni
Senior Principal Software Engineer
Functional Safety
In-vehicle OS

Agenda

- How we ended up here?
- Why documentation is important
- What are the aspects to be documented
- What we have in Linux today and how it maps to the different aspects
- Potential improvements and how to involve more experts in writing and maintaining the documentation

How we ended up here?

Safety Architecture WG

This presentation is based on a work-in-progress document that the Safety Architecture working group in ELISA is working on.

If you wish to join the discussion, sign up here: https://lists.elisa.tech/g/safety-architecture

ELISA
Enabling Linux in Safety Applications

LINUX PLUMBERS CONFERENCE | Vienna, Austria Sept. 18-20, 2024

Why documentation is important…

image from https://www.raintels.com/uploads/casestudies/17-sqa.jpg

The core principles behind Software quality are:

- Defining the expected behavior of the code and
- Providing sufficient verification evidences against it

Why documentation is important…

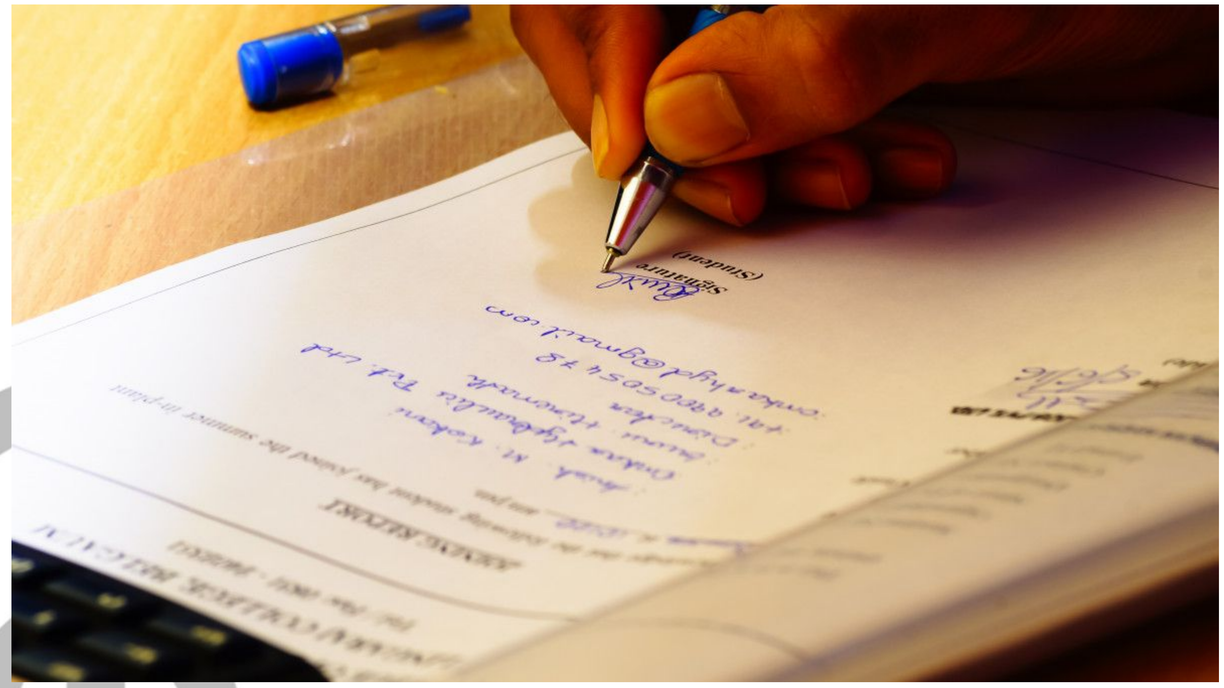

image from https://pxhere.com/en/photo/608137

How does documentation support the quality key principles?

Main Objectives:

- Integrators (i.e. the users of Linux) need to rely on documentation to understand the expected behaviour of the code and assess it against their requirements
- Developers need to write patches in compliance with the documented expected behaviour (or else they also need to update the documentation of it)
- Testers need to design and run verification measures on the basis of the documented expected behaviour (as running verification measures based on the code is expensive and it may lead to biased results)

What are the Software Architectural design aspects to be documented?

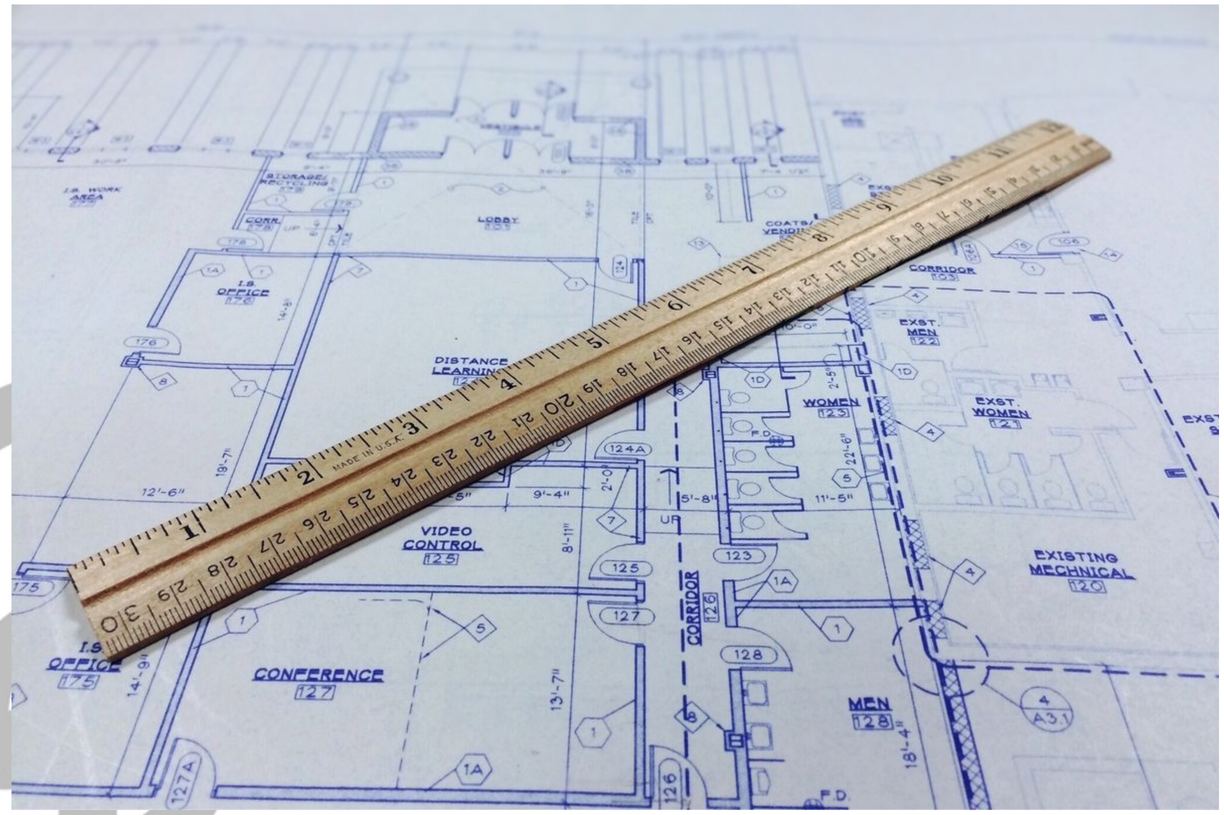

image from https://www.8base.com/blog/saas-api

In order to meet the main objectives from the previous slide, most of quality and safety standards demand the following SW Architectural design aspects to be documented

Static Design Aspects:

- Main Components and requirements/specifications allocated to them
- SW/HW interfaces
- SW/HW resources

Dynamic Design Aspects:

- chain of events/behaviour
- the logical sequence of data processing
- control flow and data flow
- temporal constraints

And…any **compile time or runtime configuration parameter** impacting the above mentioned aspects

What Linux provides WRT such aspects?

Static Design Aspects:

- **Main Components and requirements/specifications allocated to them.**
  - Integrator's POV - From an integrator perspective the main components are the interfaces exposed to the user. The relevant doc is:
    - The Linux Manpage project:
    - The Linux kernel user-space API guide
    - The Linux kernel user's and administrator's guide
    - The GNU C Library Reference Manual

    Exported Kernel symbols shall be documented following the guidelines in "Writing kernel-doc comments"

  – Developer's POV:
    The main Kernel components are defined by the MAINTAINERs file (drivers/subsystems)
    The "most relevant" APIs can be documented following the guidelines in "Writing kernel-doc comments"

    cregit is a web based tool that can be used to retrieve commits associated with a set of code lines and their respective mailing list discussions (that also include the cover letter, if there is one).

What Linux provides WRT such aspects?

- **SW/HW interfaces:**
  - Integrator's POV - covered in the "Main Components" section
  - Developer's POV - dependencies between subsystems are lacking documentation today (AFAIK). A possibility to effectively parse and visualize dependencies is the ks-nav tool.

    Firmware interfaces are documented in "Documentation/devicetree/bindings" and in the ACPI specifications for devicetree and ACPI based FW respectively

- **SW/HW resources**
  - Integrator's POV - covered above in the "Main Components" section. Relevant resources are documented as part of the user documentation
  - Developer's POV - From a developer perspective relevant SW/HW resources can be documented following the members documentation template, however such template does not enforce specifying which subsystem or driver uses them. In order to fill this gap the ks-nav tool can be used

What Linux provides WRT such aspects?

Dynamic Design Aspects:

- chain of events/behaviour
- the logical sequence of data processing
- control flow and data flow
- temporal constraints

Unfortunately today all aspects above are not enforced from a Documentation template point of view. These aspects can be documented, especially in the Overview sections or in the drivers' or subsystems' specific RST files, however there are no specific template fields mapping to them.

Compile time or runtime configuration parameter:
TODO: We still need to analyse this aspect in the ELISA working group.

What Linux provides WRT such aspects?

Summary

There are tools that can be used to document or reverse engineer the code to cover:

- Requirements/ Specifications
- Static Design Aspects
- Dynamic Design Aspects (even if the template is not optimal)

However

Are Drivers/Subsystems consistently documented?

(Open Discussion) How to involve more expert? How to better meet the Software Architectural Design Objective?

image from https://printerval.com/

DISCLAIMER!!!
We are not trying to enforce the Linux community to cover all documentation aspects listed above, but instead we are trying to find a common ground where documentation can be improved on some of such aspects and where such improvements are acceptable to maintainers

Some points think about:

- Tools to warn a contributor to also push a documentation update and where it would be reasonable to do so (maybe patching checkpatch?)
- What is the sync process between Kernel internal changes and the manpage or glibc manual?
- Can we revisit cregit to have an option to parse right away the patch series covers letters associated with a certain function (and sort in chronological order)?
- Is there any change we can make to the Overview template to enforce some architectural/design aspects?
- Any other reasonable change that you can think of?

Linux Plumbers Conference

Vienna, Austria | September 18-20, 2024