Embedded Linux Conference Europe

OpenIoT Summit Europe

Collaborate on Linux for Use in Safety-Critical Systems

Lukas Bulwahn, BMW Car IT GmbH

THE LINUX FOUNDATION

Who am I

- Lukas Bulwahn, Functional Safety Software Expert at BMW Car IT GmbH
- Working at BMW Car IT GmbH since 2012:
  - since 2012: Researching on an Open-Source Software Platform for Autonomous Driving
  - 2014 - 2016: Contributing to the definition of the Adaptive AUTOSAR Platform in the Open Industrial Collaboration AUTOSAR
  - since 2016: Contributing to the safety argumentation of the SIL2LinuxMP collaboration project at OSADL

- Presenting here in three roles:
  - Contributing Member of the SIL2LinuxMP collaboration project at OSADL
  - Contributing Member in the Linux in Safety-Critical Systems formation group, headed by Kate Stewart at The Linux Foundation
  - Employee at BMW, contributing to the operating system safety argumentation in autonomous driving

Motivating Linux in Safety-Critical Systems

Business Motivation

THE WALL STREET JOURNAL.

ESSAY

Why Software Is Eating The World

By Marc Andreessen

August 20, 2011

Main message: Software innovations will disrupt every industry, even very established industries.

Companies in the mechatronics industry are struggling with this change which is being driven by software innovations and software industry competitors.

Give profits to the software vendors or invest to explore alternatives?

THE LINUX FOUNDATION

The History of UNIX

1980s

Various UNIX operating systems in the market. Vendors largely controlled their users, with a vendor-lock-in incompability mess.

1990s

Linux was born. Linux was formed as the coalition of those tired users and losers.
With an open-source collaboration model, Linux built a strong ecosystem of users and software companies.

Today

Linux (in 2017):

- 90% public cloud workload
- 62% embedded market share
- 99% supercomputer market share
- 82% smartphone market share

(Source: https://www.linuxfoundation.org/publications/2017-state-of-linux-kernel-development/)

Facebook, Google, IBM, Intel... have Linux kernel teams for their Linux-based operating systems.
They control the software chain & stack, although they sell hardware and services rather than software.

History repeats itself

Mechatronic industry is now at the same crossroads for safety-critical operating systems to run complex algorithms and software.

How to create a healthy ecosystem of safety-critical operating systems and focus on innovative software functions?

Pro's and Con's of Linux

The Linux kernel has:

- Large Development Ecosystem
- Security Capabilities
- Multi-Core Support
- Unmatched Hardware Support
- Many Linux Experts at all levels available

The Linux kernel is missing:

- Hard Real-time Capabilities
- Proven Safety-compliant Development Process

Can these gaps be closed?

THE LINUX FOUNDATION

OSADL SIL2LinuxMP Project

- **Mission:**
  - **Provide procedures and methods** to qualify Linux on a multi-core embedded platform at safety integrity level 2 (SIL2) according to IEC 61508 Ed 2.
  - **Show feasibility** of **procedures and methods** on a real-world system
  - **Show potential** for collaboration and re-use of Linux kernel analysis

- **Collaborative project of industrial & research partners**
  - Project running since 2015, organized by OSADL
  - Full members: BMW Car IT, Intel (since ‘17), A&R Tech, KUKA, Sensor-Technik Wiedemann (full members till ’16, reviewing members in ‘17)
  - Reviewing members: Bosch, Elektrobit, Hitachi, Linutronix, MBDA Italia, MEN Mikro Elektronik, Mentor, OpenSynergy, Pilz GmbH & Co. KG, Renesas, Vienna Water Monitoring Solutions
  - Academic members: A. Khoroshilov (ISP RAS), K. Chow (Lanzhou Univ.), J. Lawall (Inria/LIP6), F. Tränkle (HS Heilbronn)
  - Experts from certification bodies: B. Nölte (TÜV Süd), O. Busa, R. Heinen, H. Schäbe (TÜV Rheinland)
  - SIL2LinuxMP core working team: N. McGuire, A. Platschek, L. Böhm, M. Kreidl (OpenTech)

Introduction to Functional Safety

THE LINUX FOUNDATION

Functional Safety

"**Functional safety** is the part of the overall safety of a system (...) that depends on the system (...) **operating correctly in response to its inputs**, including the safe management of likely **operator errors, hardware failures and environmental changes**.

The **objective of functional safety** is freedom from **unacceptable risk** of physical injury or of damage to the health of people either directly or indirectly."

(Source: wikipedia.org:Functional Safety)

- Work on Functional Safety is **Risk Management**
  - Risk Management is to **focus quality assurance on the right aspects and right parts!**
  - It is NOT to do just more work or write hundreds of documents!

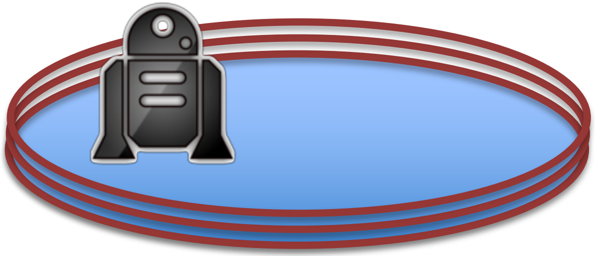
Functional Safety

- How to determine the **acceptable risk?**
  - Agreement in **global safety standards**
  - **IEC 61508**: Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems - a basic functional safety standard applicable to all kinds of industry
    - Adaptations to specific industries: ISO 26262 for automotive, IEC 62279 for railway applications

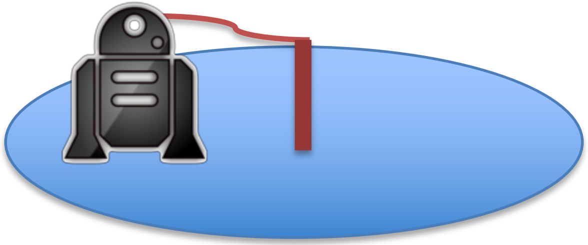
- How to **design safe systems?**
  - **System design & system analysis**
    - Analyze your system to know which parts must be of high quality for the system's safety
    - Assign safety integrity levels (SIL) to those parts, SIL1 (low safety level) to SIL4 (high safety level)
  - **Rigorous development process**
    - Develop those parts with high SIL with sufficient rigor (= the right development process)
    - Safety standards state which objectives shall be achieved in each development phase

Functional Safety by Example I

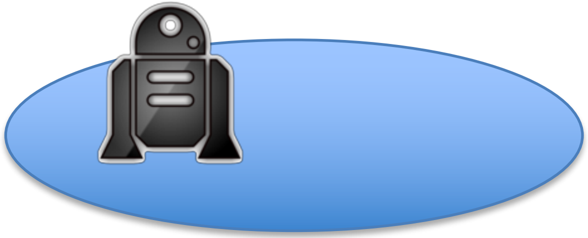
The robot can only harm somebody when it leaves the blue area.



Is this system safe?



Is this system safe?



Is this system safe?

To assess whether your system is safe, you need to understand your system sufficiently.

THE LINUX FOUNDATION

Functional Safety by Example II

The robot can only harm somebody, when the steering application steers the robot to leave the blue area.

Steering Application
glibc
Kernel
Hardware

Is this system safe?

Steering Application
glibc
Kernel
Hypervisor
Hardware

Is this system safe?

To assess whether your system is safe, you need to understand your system sufficiently.

Linux in Safety-Critical Applications

Safety-Critical Linux I

To assess whether your system is safe,
you need to understand your system sufficiently.

⇒ If your system‘s safety depends on Linux,
you need to understand Linux sufficiently for
your system context and use

Safety by Process Argumentation

Compliance to Objectives of Safety Standards by Development Process Assessment:

- Linux has been continuously developed for 25+ years
- Continuous Process Improvement is in place.

  When technical or procedural issues in the kernel development are identified and pressing, the community addresses them.
- Evidence for the requisite process quality and process improvement quality exists already.
- This evidence can indicate that all objectives of a safety integrity level 2 for selected parts and properties are met.

Safety-Critical Linux II

- **The difference** between Safety-Critical Linux and main-line Linux **is the way you use it**.
  - *Understand* your system and *understand* Linux
  - Make your system uses Linux based on the *selected* properties of Linux *you can assure*

How to Make Linux-based Systems Safe

Organisation‘s shared knowledge of the system and Linux makes the system safe

⇒ *Processes and Methods* to understand:
- the qualities of the complex software system
- the qualities of the Linux kernel

⇒ Education on these topics is the key to your safety product development.

More Information at *Linux in Safety Systems Summit*
(tomorrow, October 24th, 2018, 11:00 to 17:30, at Sheraton, Room Edinburgh 1;
Agenda on ELCE Collocated Events webpage)

THE LINUX FOUNDATION

SIL2LinuxMP Project in Retrospective

- Successful exchange and education of challenges and ideas:
  - A defined plan and compliance route
    - Reviewed by project participants and a safety authority
  - Some first technical investigations:
    - System engineering methods for complex software systems
    - Methods and tools for kernel investigations and gathering process evidence
    - Understanding existing Linux kernel verification tools

Things to Keep

- **What was important and what went well?**

  Education and exchange of ideas in eight three-day workshops

  - System safety engineering for complex software system
  - Interpretation of the IEC 61508 for pre-existing software
  - Relevant verification tools applied in the Linux kernel development

Lessons Learned and Issues

- Organised as research project, not as collaboration
- Underestimated collaboration around functional safety
  - Difficult & mind-bending field, different from software engineering
  - Open Collaboration in functional safety was not established
- Misunderstanding of educational goal
- No suitable hardware and access to documentation suitable for collaboration
- Members with little participation had difficulties to make use of results

From Research to Collaboration

Goal of a Collaboration

- Shared development and effort on:
  - Understanding safety engineering of complex systems
  - Creating risk assessments of the kernel subsystems and features
  - Gathering evidences of kernel development process compliance
  - Developing supporting tools
  - Creating material to train and educate engineers

Central Elements of a Collaboration

- Establish well-defined governance and project steering in a neutral organisation
- Maintain good community health
- Keep educating on functional safety and process assessment
- Share a common system to focus on common activities
- Reach out to Linux and safety communities, and to hardware vendors

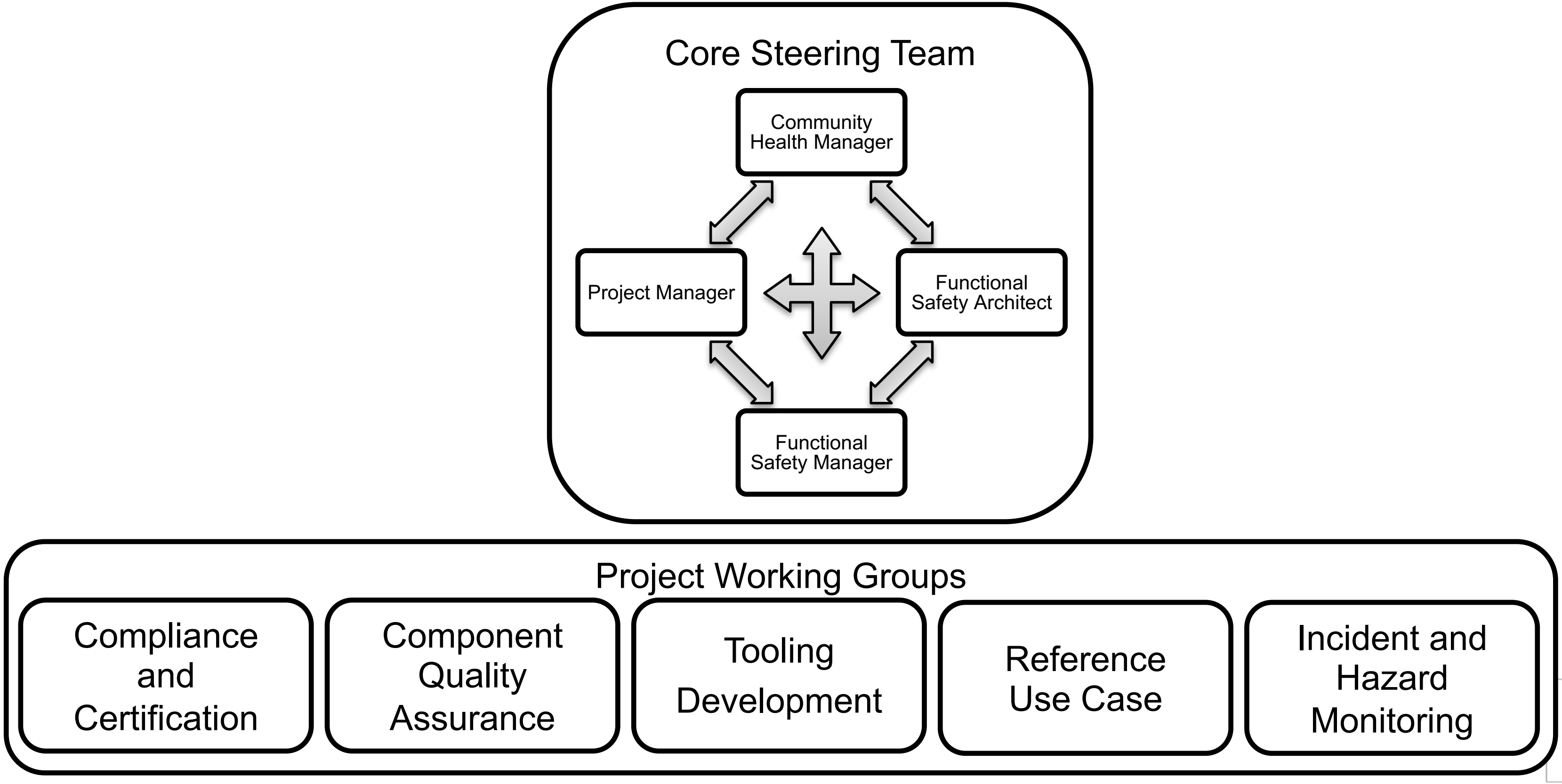
Successful Outcome of a Collaboration

Assets for safety certification of Linux-based systems

- consisting of a **complete process**, selected kernel features and tools, and previous process assessments
- **shown feasible** with a reference system
- **usable** by properly educated system integrators
- **maintained** over industrial-grade product lifetimes
- **well-known and accepted** by safety community, certification authorities and standardization bodies in multiple industries
- **positively recognised** and impacting the Linux kernel community
- **with hardware collateral** from multiple supporting vendors

THE LINUX FOUNDATION

Project Organisation



Limits of Collaboration

- The collaboration:
  - *cannot engineer* your system to be safe
  - *cannot ensure* that you know how to apply the described process and methods
  - *cannot create* an out-of-tree Linux kernel for safety-critical applications
    (Remember the continuous process improvement argument!)
  - *cannot relieve* you from your responsibilities, legal obligations and liabilities.

Modes of Collaboration

- Modes of Collaboration:
  - Informal exchange of experts
  - Common training for needed activities
  - Shared development of tools
  - Shared maintenance of evidences on Linux development
  - Collaboration on a specific use case

Risks and Opportunities

- Risks and Challenges:
  - Mixed track record of open-source projects in the embedded domain
  - No known examples for collaborative safety engineering activities
  - Little technical contributions lead to a large project setup without proper contribution
  - Provision of reference hardware and relevant functional safety documents to contributors is impossible
  - Analysis of processes is too involved to establish or maintain
- Opportunities:
  - Community accepts and supports our initiatives on quality and development processes
  - kernel/glibc development processes adjust due to our findings

- Different conceptual approaches
  - can result in an overall more robust argumentation
  - can lead to thinking in „camps“ beyond the actual objective assessment

Conclusion

Conclusion

- Industry needs an operating system for complex algorithms and software suitable for safety-critical systems
- Basis available:
  - Functional safety is about managing risk in product development
  - Risk in Linux-based systems are only understood with knowledge of the system and kernel
  - Basis for understanding Linux in safety-critical systems available
- Collaboration proposal:
  - Further development of the basis requires industry collaboration
  - Technical and organisational proposal is in place

The future of Enabling Linux In Safety Applications is up to all of us...

More at Linux in Safety Systems Summit

(tomorrow, October 24th, 2018; 11:00 - 17:30 at Sheraton)

Agenda on ELCE Collocated Events webpage

Thank you for your attention!

Embedded Linux Conference

Europe

OpenIoT Summit

Europe

THE LINUX FOUNDATION

Copyright on this Presentation