OSADL SIL2LinuxMP - Linux Safety Qualification

Nicholas Mc Guire <safety@osadl.org>

October 11, 2014

Outline

- Context
- Justifying GNU/Linux
- A bit on Tools
- Conclusions

Why Linux for Automotive Safety

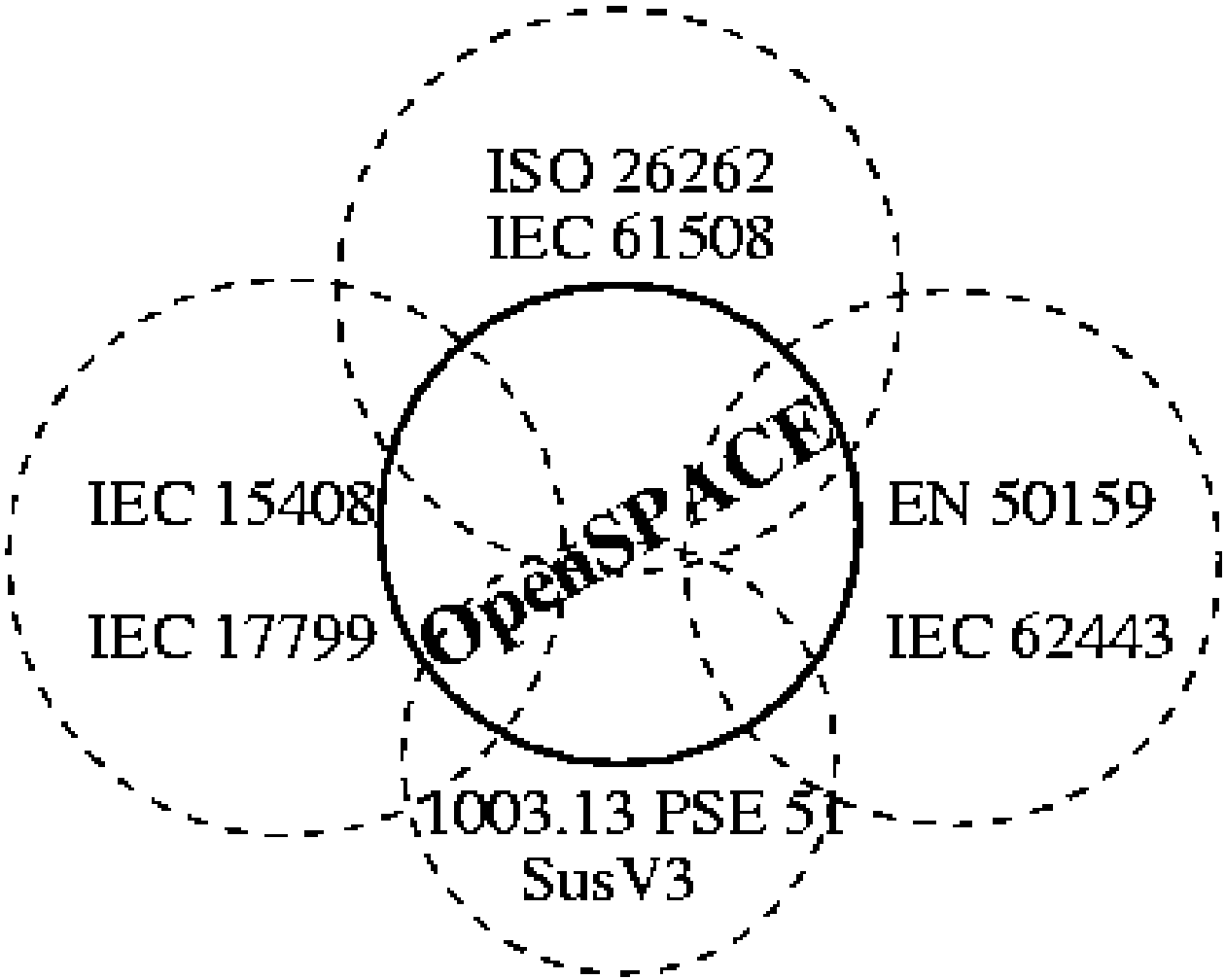
Open Source Automation Develo

Why Linux for safety ?

- Satisfy Demands:
  - General security demands
  - Performance demands in cognitive systems
  - Functional requirements of diagnostics and autonomous systems
- Satisfy Certification
  - Standardization
  - Established concepts
  - Breadth of deployment
  - Development model

Using well-selected FLOSS for safety has clear technical advantages - developing the reusable generic arguments is a key element of SIL2LinuxMP.

Standards Context



Complex software and Safety

"As far as practicable the design shall keep the safety-related part of the software simple" [61508-3 Ed 7.4.2.6]

"This Standard does not discourage the use of software in safety-critical systems. When designed and implemented correctly, software is often the first, and sometimes the best, hazard detection and prevention mechanism in the system." [NASA NPR 8719.13B 1.2]

OSADL SIL2LinuxMP - Linux Safety Qualification

Nicholas Mc Guire <safety@osadl.

High-level safety process outline

- Defined organization - OSADL
- Develop use-case - autonomous driving
- Perform hazard analysis - focus on BH-Safety "generic functions OS"
- Derive risks and risk reduction needs bounded by ASIL B
- Apply suitable/accepted methods - mitigate residual risks
- Specify tool set/work-flow and tools assurance procedures
- Justify process,methods and evidence - safety case template

Organizational Framework

- Organization: Open Source Automation Development Lab
- Planing: OSADL Safety Critical Linux Working Group
- Execution: participating industrial and academic OSADL members
- Technical Infrastructure: OSADL in Heidelberg
- Certification Authority: TueV Rheinland

Justification Strategy

- Develop/communicate and agree on the terminology
- Define the overall safety life-cycle strategy
- Develop the system context in ISO 26262 SEooC + "Qualification of software components"-> IEC 61508 system context
- Develop the software context in 61508-3 (selection)
- Select methods 61508-5/6/7 and other standards
- Develop FLOSS/Linux specific methods where needed
- Embed in well specified process, trace/report -> justification

Generic Strategy PIU

Strategy development example "proven-in-use"

```
61508-1
  61508-2 7.4.2.2
          `-> 7.4.10
              `- 7.4.10.1-7
                 `-- 61508-7 B 5.4
                              `-- C 2.10
```

Extract argument "templates", argument structure and assurance methods.

Generic Strategy pre-existing SW

Most (all ?) of the current functional safety standards describe the concept of pre-existing SW

- IEC 61508 Ed 2 (Generic)
- EN 50128 Ed 2 (Rail)
- ISO 26262 (Automotive)

> `pre-existing software`
> `software developed prior to the application currently in question,including COTS ( commercial-off-the-shelf) and open source software.`

[EN 50128 Ed 2 2012 Clause 3.1.17]

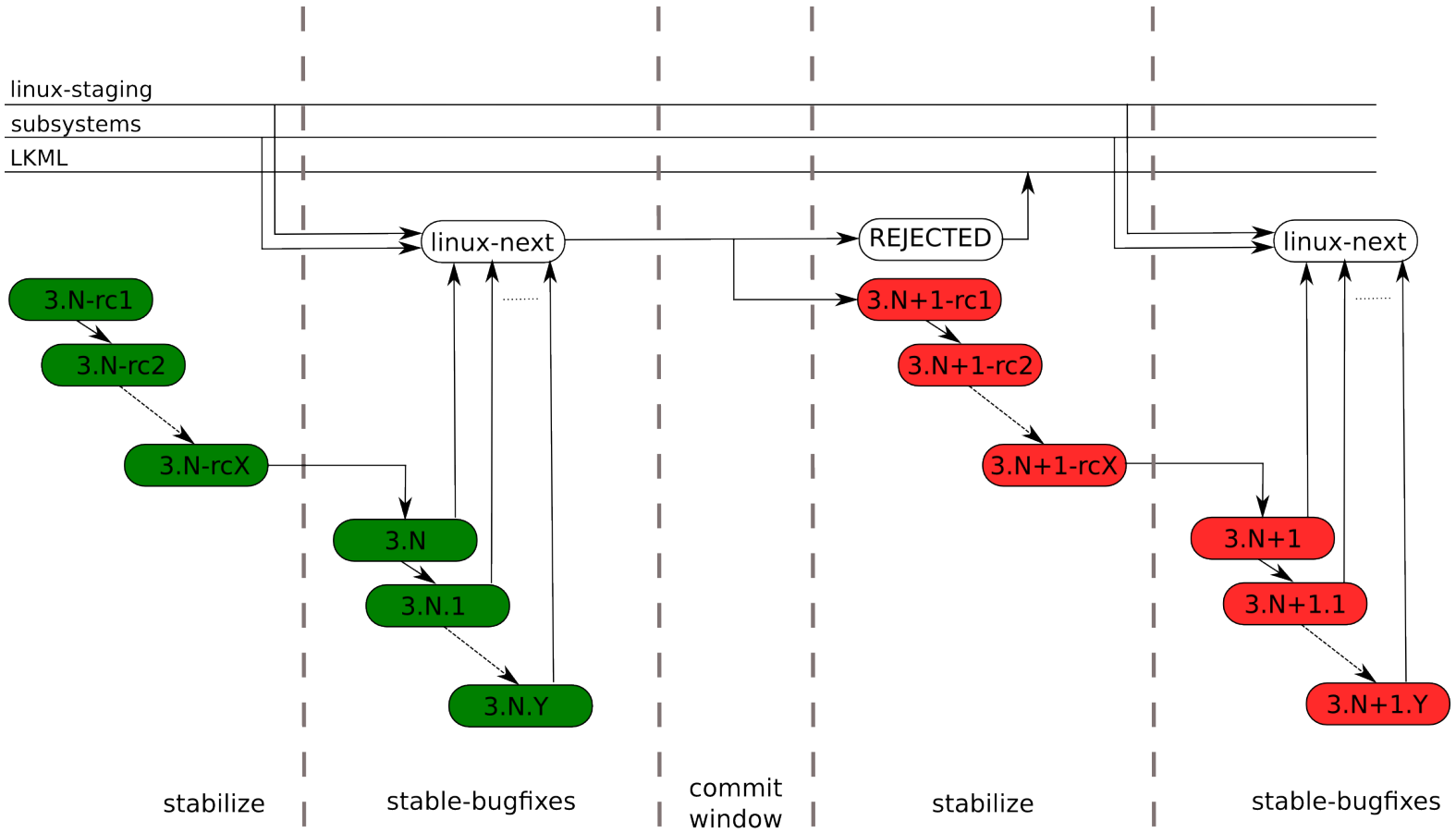
Generic Strategy - ISO 26262

Strategy development example in the context of ISO 26262 pre-existing + Safety Element out-of Context (SEooC)

```
GNU/Linux as SEooC ISO 26262-10 Clause 10
‘-> ISO 26262-8
    +-> 12.3.1 Prerequisites
    |   +-> Pre-determined ASIL -> ASIL B
    |   ‘-> Requirements known -> POSIX subset
    +-> 12.4.3.1 The specification of the SW
    |   ‘-> SuSV3,POSIX,1003.13
    +-> 12.4.3.2 evidence of compliance
    |   +-> posixtestsuit
    |   ‘-> OSADL QA-Farm data
    ‘-> 12.4.4.1 qualified for different domain
        ‘-> IEC 61508 Route 3S
```

Map to existing requirements, specifications...methods.

Linux DLC Qualities

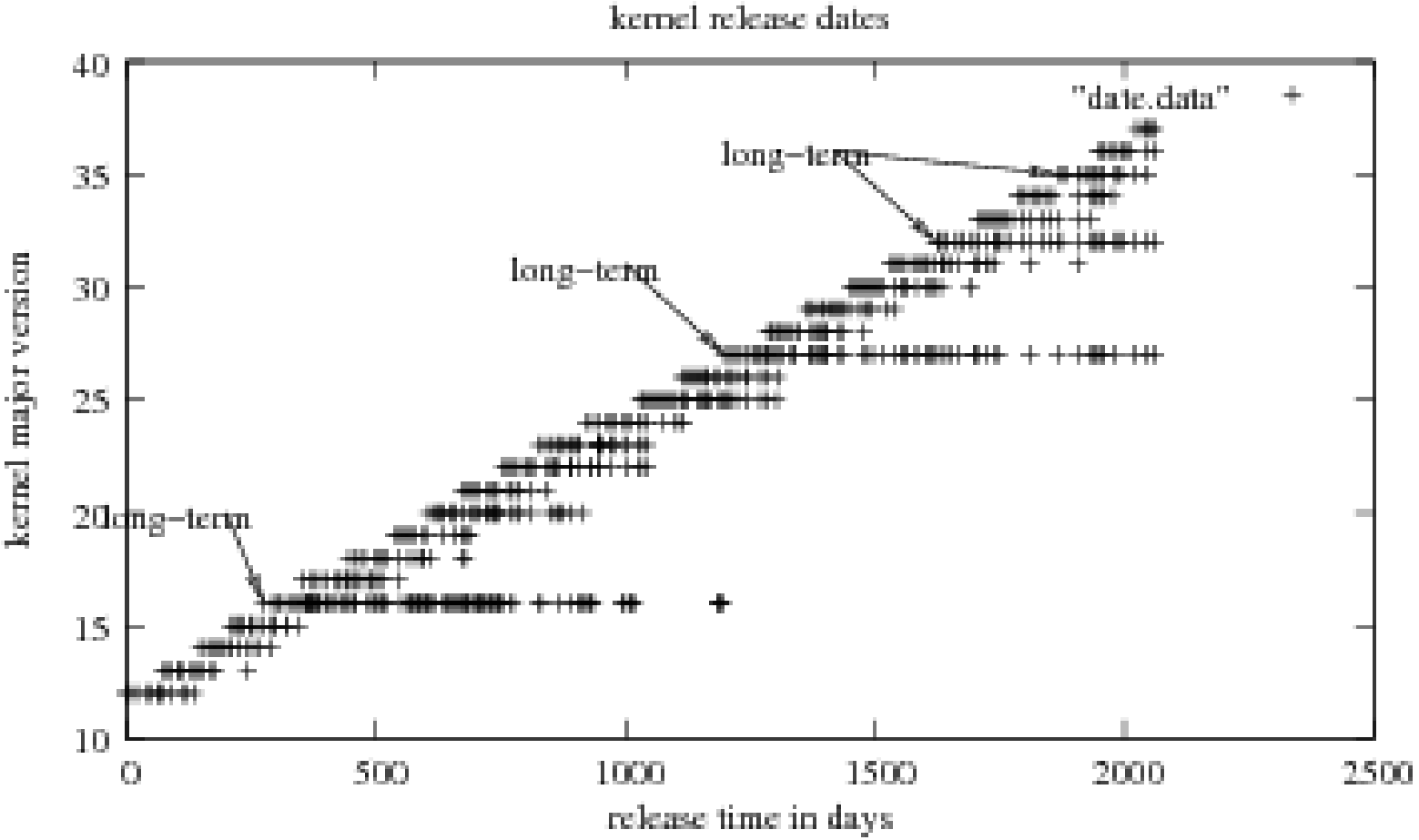


Assessment of procedures

- CodinStyle
- SubmittingPatches/SubmitChecklist
- Changes - minimum tool requirements
- MAINTAINERS - Well defined Roles

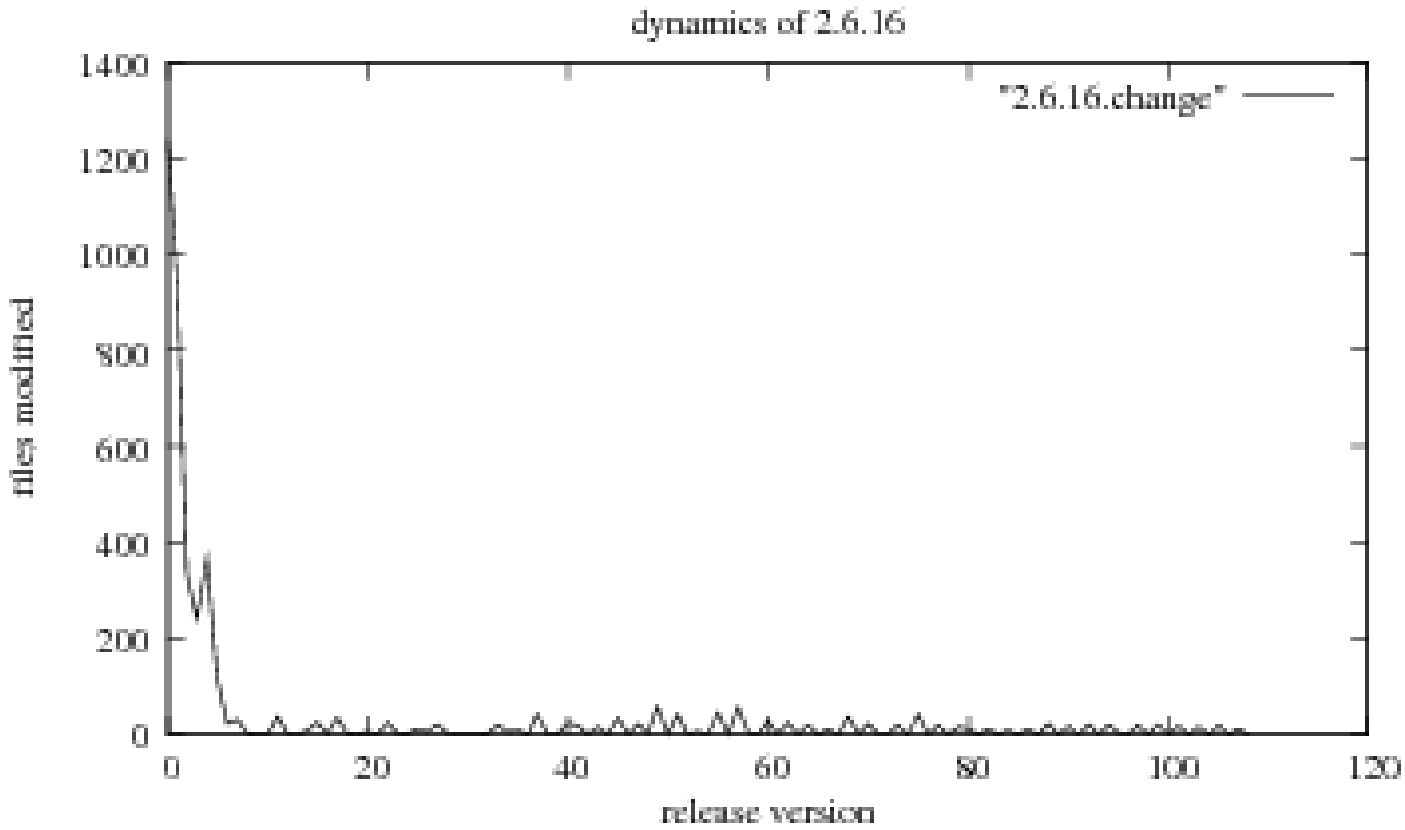
Arguing the Linux DLC and performing a gap-analysis will be needed - the key issue is that the DLC of the Linux kernel is in fact rigorous and robust.

Suitability of version



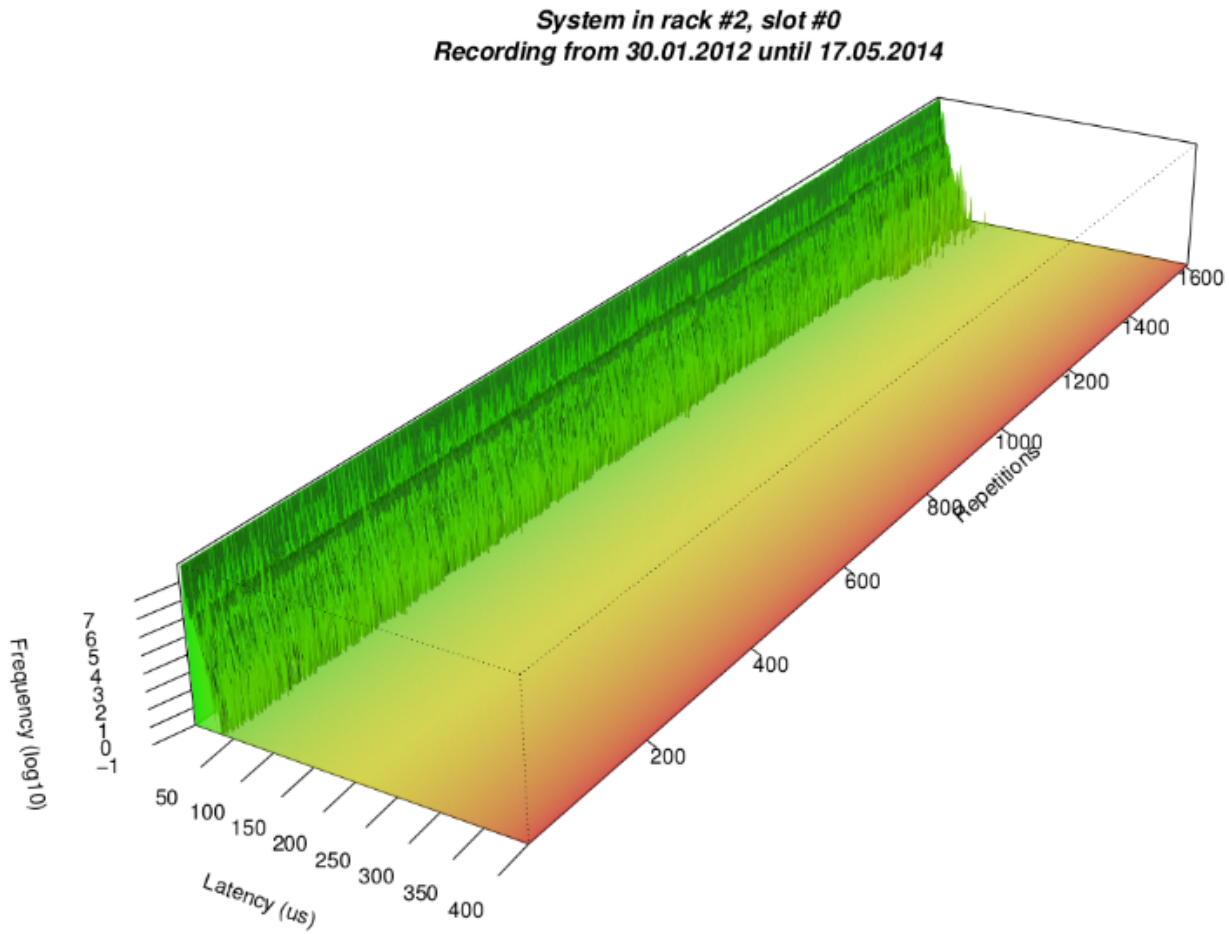
A well defined selection process backed by data

Arguments and Methods



Monitoring of GNU/Linux DLC: determine relevance and conditional impact analysis,

Assurance Data - OSADL QA-Farm



Long term measurements under specified conditions - http://www.osadl.org/QA

OSADL SIL2LinuxMP - Linux Safety Qualification

Nicholas Mc Guire <safety@osadl.

Tools Classification

Qualification levels of tools (incomplete preliminary list)

- T3: gcc, sparse, coccinelle, checkpatch.pl(?)
- T2: lockdep,ftrace,rt-tests
- T1: LTP subset (?), git, rmtoo,etc.

The primary management could be rmtoo via git

Tools Qualification Methods

- T1/T2/T3 definition of monitoring process (periodic, reports)
- T1/T2/T3 testing in context of selected distribution
- T2/T3 analysis of available data == "increased confidence from use"
- T2/T3 definition of subset to be used
- T2/T3 assessment of DLC

Methods: rigor R1 (for all) R2 for code emitting tools (T3) if doable

Conclusion

- Most elements are there - selection and integration needed
- Linux for Safety related systems at ASIL B is doable
- Early coordination of proposed qualification route with functional safety experts from the automotive domain / certification authority needed
- The necessary tools can be provided as open-source set allowing a broad usage of processes, procedures and methods
- The technical/functional demands of autonomous driving systems can be covered by GNU/Linux RTOS

Nobody claims this will be simple - but we do not see it as high-risk at this point.

Thanks!