EVALUATION OF OPEN SOURCE OPERATING SYSTEMS FOR SAFETY-CRITICAL APPLICATIONS

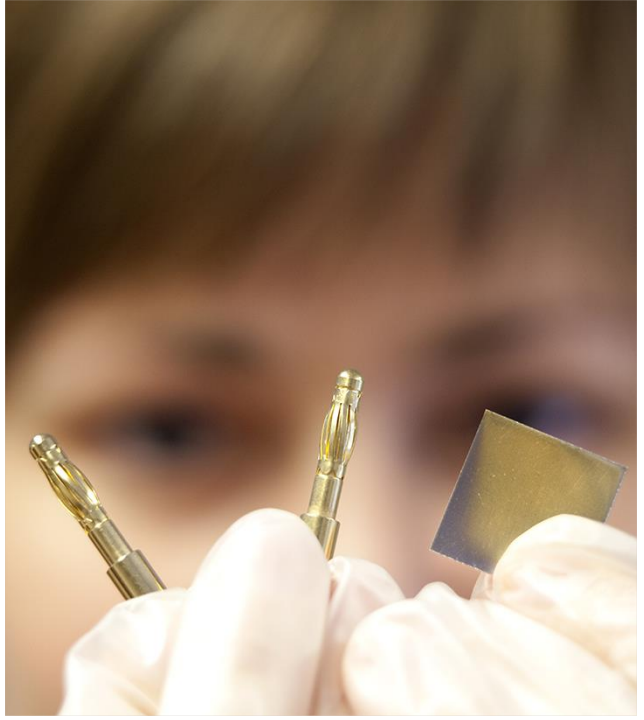
Petter Sainio Berntsson, Lars Strandén, Fredrik Warg

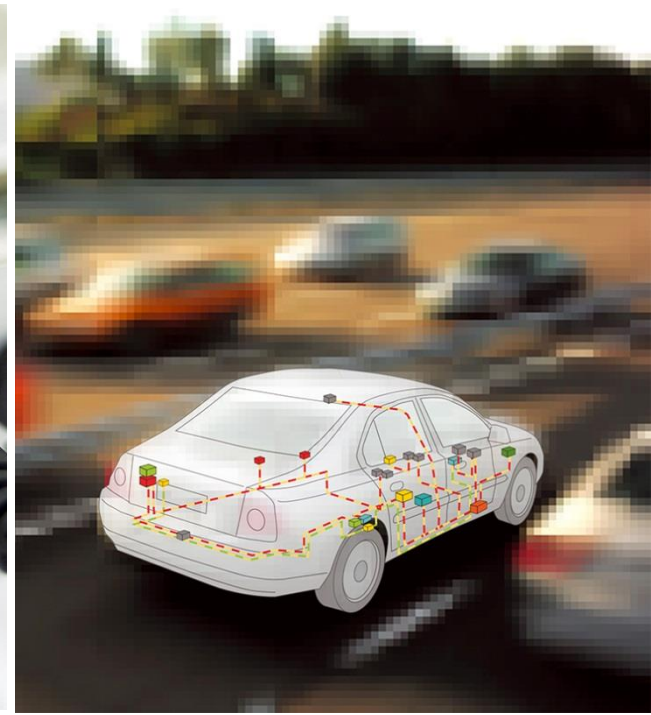
Presenter:
Fredrik Warg, Senior Researcher - Dependable Systems

SERENE 2017 - 9th International Workshop on Software Engineering for Resilient Systems
Geneva, Switzerland, September 2017

Research Institutes of Sweden

SAFETY AND TRANSPORT
ELECTRONICS





Our question is…

Can we use an open source real-time operating system in a safety-critical application, and if so under which conditions?

- Why are we interested?
  - RISE works with research, verification (testbeds, technical assessments), certification
  - In the Dependable Systems group, we work with functional safety standards such as ISO 26262 (Road Vehicles) and IEC 61508 (E/E/PE safety related systems)
- We see increasing interest in OSS components from customers and partners
  - Can be e.g. OS, libraries, development tools. Here we focus on RTOSs.

Some terms

- Open source software (OSS)
  - Software with a license allowing anyone to study, change, and freely re-distribute the software
    - No license fees (though commercially supported versions may have)
    - Possible for user to make own modifications
  - Typically uses a collaborative software development model
    - Developed publicly by a distributed group of developers and users communicating over the internet
- Real-time Operating System (RTOS)
  - Used when correctness does not only depend on the correctness of the results but also whether the result is produced within a set time constraint (deadline)
  - Common in embedded and safety-critical systems

Open source software and functional safety standards

- Functional safety
  - Part of overall safety relating to E/E systems
    - Freedom from unacceptable risk of harm
  - Functional safety standards typically prescribe a process including
    - Hazard analysis and risk assessment
    - Managed lifecycle for ensuring safety functions performs to the design intent
    - Strict verification and validation activities
    - Documentation of argument and evidence that standard and design intent are met (the safety case)
    - Audits and assessments
- Large OSS projects often known to have ”good quality”
  - But perhaps mainly quality attributes such as user satisfaction, performance, features, compatibility?
  - Typically not developed with safety-critical systems in mind
- How does a typical OSS project compare to the requirements of functional safety standards?

Possible conflicts between OSS and requirements for safe software

(Theo Jacobs, Fraunhofer, 2016)

Requirement/Recommendation		***Reality in OSS development***
▪ Risk assessment considers a certain application with defined boundary conditions	≠	▪ Re-use of code in various software projects with different application and conditions
▪ Creation of validated software versions, Protection against uncontrolled modification	≠	▪ Free alteration, recombination and forking, No control for the developer who uses the software
▪ Management of the complete software lifecycle with clear responsibilities and full documentation	≠	▪ Loosely connected development teams, limited trust between developers, varying quality of documentation
▪ Legal responsibility of a single entity (manufacturing company), contracts with suppliers	≠	▪ End user is responsible for the complete software, no legally binding agreements with developers

http://rosindustrial.org/events/2016/11/3/2016-ros-industrial-conference

RI. SE

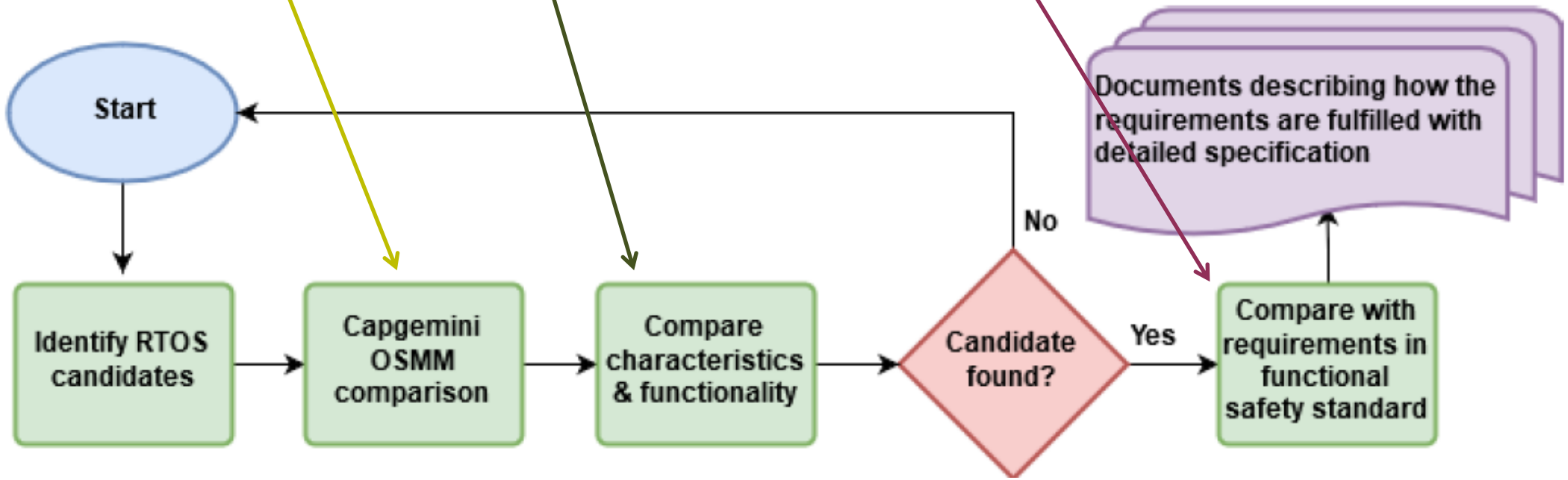
Our question – refined…

Can we use an open source real-time operating system in a safety-critical application, and if so under which conditions?

- How to evaluate the quality of an open source project in general?
- How to evaluate the suitability (feature-wise) of an RTOS for use in safety-critical applications?
- Can the software be qualified for use according to functional safety standards?

Proposed workflow for RTOS evaluation and qualification

- Answers each of the three mentioned questions in turn:
  - Quality of the project
  - Suitability feature-wise for a safety-critical system
  - Possibility to quality for the intended functional safety standard
- Aim to reduce work/risk by evaluating candidates before spending effort on qualification towards a standard



General assessment of product and process quality for OSS

- Objective: Determine if the OSS project is suitable with respect to issues such as…
  - Overall quality - good enough to rely on for the intended use?
  - Focus on issues such as maturity, availability of support, market penetration, usability etc
- Our proposal: Use software quality metrics
  - Standardized quality model exists - ISO 25010 Systems and Software Quality
  - However, some attributes especially important for OSS, such as judging the impact of community, collaboration, and support availability aspects, are not obvious
- We have used the Capgemini open source maturity model (OSSM)
  - Similar to ISO 25010 but suitable for open source projects
  - Product indicators – Objective and measurable fact about the product (context independent)
  - Application indicators – How well the product fits a specific context

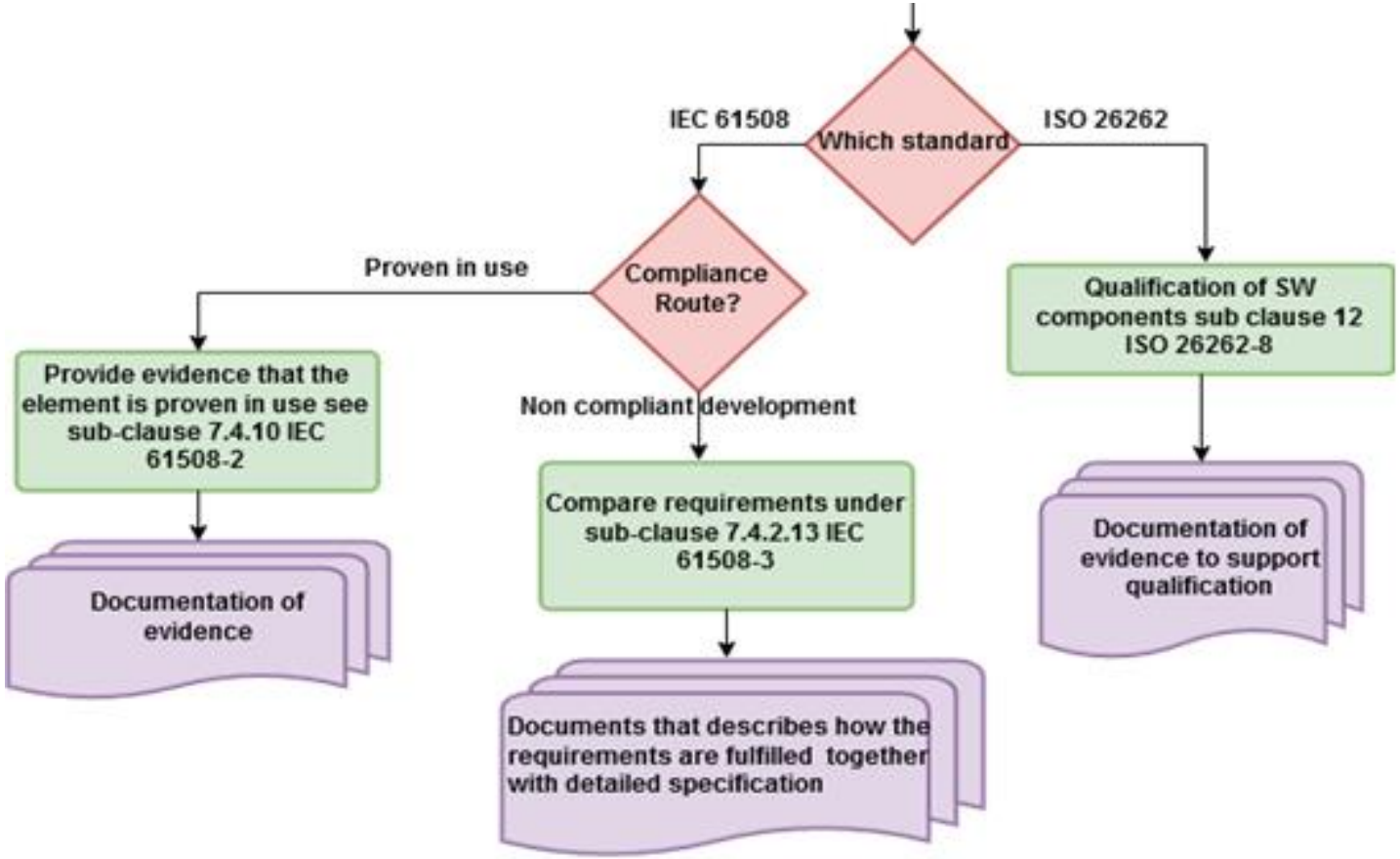
Safety-relevant characteristics for an RTOS

- Objective: Determine if the feature-set of an RTOS is suitable for safety-critical applications
  - Find issues such as domain-separation, use of coding rules and language subsets, real-time properties etc that are especially important for safety-critical applications and often mandated by standards
- Our proposal: Create an evaluation model for safety-relevant characteristics
- We have assembled a first version of a checklist based on standards and a literature survey (shown in example)
- Meant to be an indicator to the suitability and amount of effort might be needed to adopt a specific RTOS in a safety-critical system

Compliance against relevant functional safety standard

- Objective: Determine compliance against a particular functional safety standard
  - Match the RTOS OSS against relevant requirements in a particular standard
  - Determine shortfalls– are they show-stoppers or possible to supplement? At what cost

- This step is dependent on both the safety-critical application and specific to the standard to be used.
- More labor intensive than previous evaluations as we must dig into more detail.



IEC 61508 compliance: Pre-existing software elements

- Sub-clause 7.4.2.12 in the IEC 61508-3 standard
- The element shall meet both requirements for a) and b) below for systematic safety integrity:
- a) Meet the requirements of one of the following compliance routes:
  - Route 1_s: Compliance with the requirements of IEC 61508.
  - Route 2_s: Provide evidence that the element is proven in use
  - **Route 3_s: Assessment of non-compliant development. Compliance with 3-7.4.2.13.**
- b) Provide a safety manual that gives a sufficiently precise and complete description of the element to make possible an assessment of the integrity of a specific safety function that depends wholly or partly on the pre-existing software element.

IEC 61508 compliance: Pre-existing software elements

- Sub-clause 7.4.2.12 in the IEC 61508-3 standard
- The element shall meet both requirements for a) and b) below for systematic safety integrity:
- a) Meet the requirements of one of the fol
  - Route 1_S: Compliance with the requirem
  - Route 2_S: Provide evidence that
  - **Route 3_S: Assessment o**
- b) Provide a safety manual that gives a su element to make possible an assessment c wholly or partly on the pre-existing softwa

- Software safety requirements specification
- Evidence that the desired safety properties have been considered
- Documentation of the element's design
- Evidence covering integration with hardware, verification, validation and review
- Evidence that unwanted functionality will not interfere with meeting safety requirements
- Evidence that failure mechanisms and mitigation measures have been implemented
- Etc.

Example: Evaluation of RTOS for IEC 61508 compliance

Evaluation according to Capgemini OSS maturity model

- We chose two candidate RTOSs suitable for small microcontrollers
  - ChibiOS
    - Compact and fast with preemptive multithreading
  - ContikiOS
    - For memory-constrained low-power applications (Internet of Things)
- Capgemini OSSM product indicators
  - Product, integration, use, and acceptance.

Indicator	ChibiOS	ContikiOS
Product		
Age	5	3
Licensing	5	3
Human hierarchies	5	5
Selling points	3	3
Developer community	3	3
Integration		
Modularity	5	5
Collaboration with other products	3	3
Standards	5	3
Use		
Support	3	3
Ease of deployment	3	3
Acceptance		
User community	3	1
Market penetration	3	1
Total	46	38

Caveat lector: *The evaluation was performed by a student (paper author P S Berntsson) as part of a Master's thesis project, and not by an expert on the Capgemini OSS maturity model, and thus should be treated as an example only.*

Example: Evaluation of RTOS for IEC 61508 compliance

Evaluation according to Capgemini OSS maturity model

- Capgemini OSSM application indicators
  - In the example we evaluate the RTOS as a component out-of-context, so the application indicators are based on an assumed generic safety-critical context

		ChibiOS		ContikiOS	
Indicator	**Priority (P)**	**Score (S)**	**P*S**	**Score (S)**	**P*S**
Usability	5	5	25	5	25
Interfacing	3	3	9	3	9
Performance	5	4	20	3	15
Reliability	5	5	25	4	20
Security	3	3	9	3	9
Proven technology	3	3	9	3	9
Vendor independence	4	4	16	4	16
Platform independence	2	4	8	4	8
Support	4	3	12	2	8
Reporting	2	4	8	2	4
Administration	2	3	6	3	6
Advice	1	1	1	1	1
Training	3	3	9	3	9
Staffing	3	2	6	2	6
Implementation	3	4	12	2	6
Total			175		151

Caveat lector: *The evaluation was performed by a student (paper author P S Berntsson) as part of a Master's thesis project, and not by an expert on the Capgemini OSS maturity model, and thus should be treated as an example only.*

Example: Evaluation of RTOS for IEC 61508 compliance

Evaluation of RTOS characteristics

- Find characteristics relevant for safety-critical systems and standards compliance
- Information mostly available in RTOS documentation
- Possible improvements
  - More fine-grained evaluation than ”yes” or ”no”
  - Further development of characteristics to match standards

Characteristic	ChibiOS	ContikiOS
Coding rules	X	X
Language subset	X	-
Version control	X	X
Documentation	X	X
Static resource allocation	X	X
Priority-based preemptive scheduling	X	-
Real-time support	X	-
Domain separation support	X	-
Synchronization primitives	X	-
Verification	X	X
Test suite	X	X
Configuration	X	X
Active community	X	X
Quality assurance	X	X
Bug tracking	X	X
Bug fixing	X	X

Example: Evaluation of RTOS for IEC 61508 compliance

Compliance to IEC 61508-3, sub-clause 7.4.2.13 route 3s for the best candidate – some highlights

- There are requirements on the architecture, for instance being modular in design, using static resource allocation, semi-formal description of architecture
  - These are fulfilled by ChibiOS and documentation is available.
- There should be evidence that elements not required for functional safety will not prevent the system from meeting its safety requirements
  - ChibiOS is modular and the work required to fulfill this requirement can be reduced by choosing the relevant functionality and excluding the rest in a particular safety-critical application.
- More detail: Petter Sainio Berntsson's Master's thesis: *publications.lib.chalmers.se/records/fulltext/249901/249901.pdf*
- Conclusion: Possible ChibiOS could be used at least for SIL 1 and 2, but additional work is needed and there are some requirements that may be more tricky to fulfill, e.g. "evidence that all credible failure mechanisms of the software element have been identified and that mitigation measures exist".

Caveat lector: *The compliance evaluation was performed by a student (paper author P S Berntsson) as part of a Master's thesis project. It is neither complete nor scrutinized by a qualified assessor, and thus should be treated as an example only.*

RI. SE

Conclusions and Future Work

- So can you use an open source RTOS in your safety-critical system?
  - Sorry, you'll get no better answer than it depends…
  - Much is context dependent (safety requirements, SIL, failure modes, standard)
  - More guidance in standards on assessing non-compliant software in general and operating systems in particular would be beneficial
- We have proposed a general strategy for evaluating if an OSS RTOS can be used in a safety-critical system
  - But our work is preliminary
  - Hopefully, there will be a continuation in some form…

RI. SE

THANKS!

Fredrik Warg

fredrik.warg@ri.se

Research Institutes of Sweden

SAFETY AND TRANSPORT ELECTRONICS