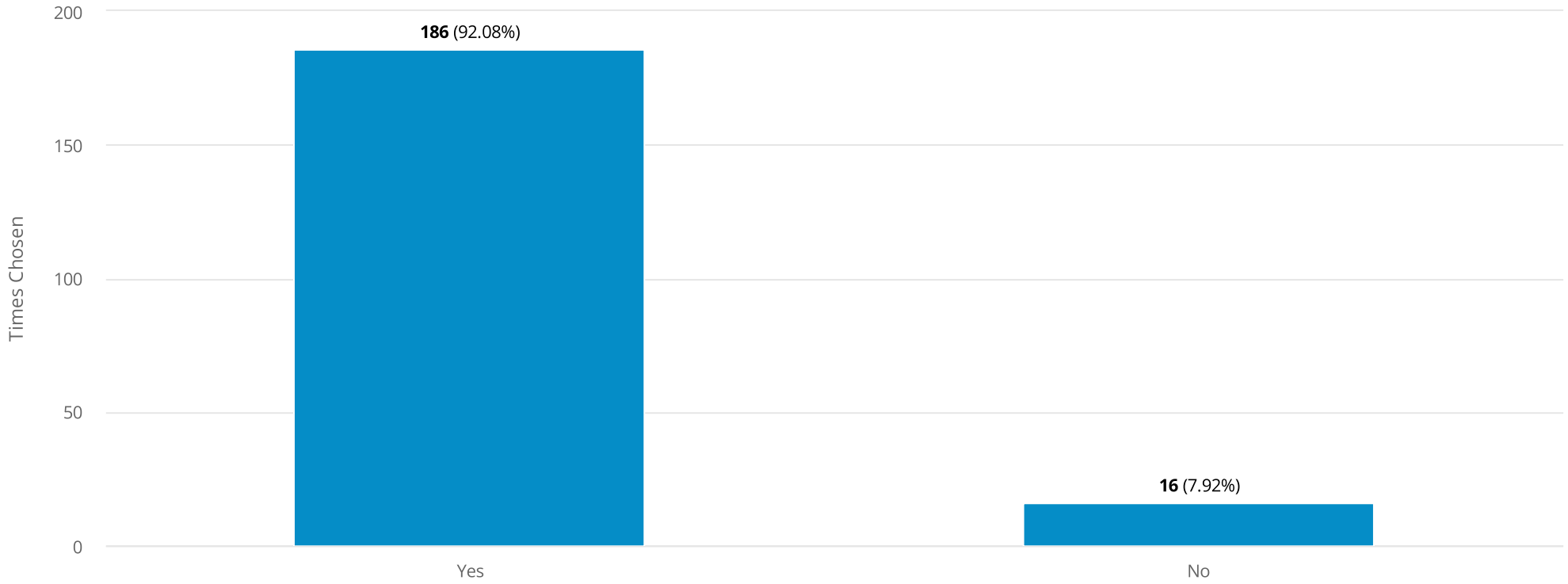
Safety-Critical Rust Adoption

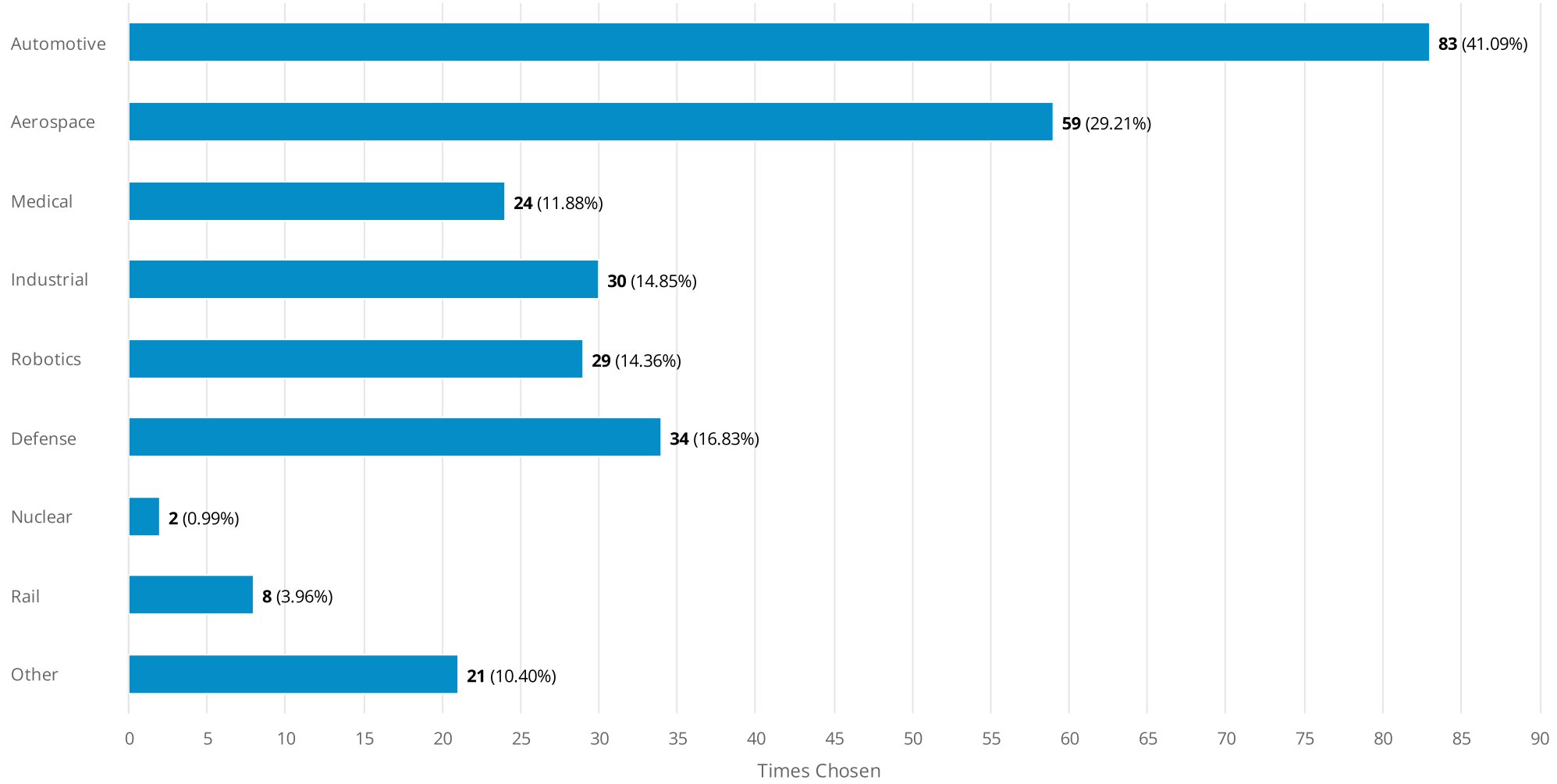
Do you work in a safety-critical industry? (Automotive, aerospace, medical, robotics, etc)

Number of responses: 202



What industry do you work in?

Number of responses: 202



"Other" text answers:

Revenue Management

Cloud
Software company
Power Systems
Finance
Underground mining
Semiconductor
Other
Semiconductor
Agriculture
Security
n/a
Maritime

marine

marketing

Advertising

Sports

Business and Finance

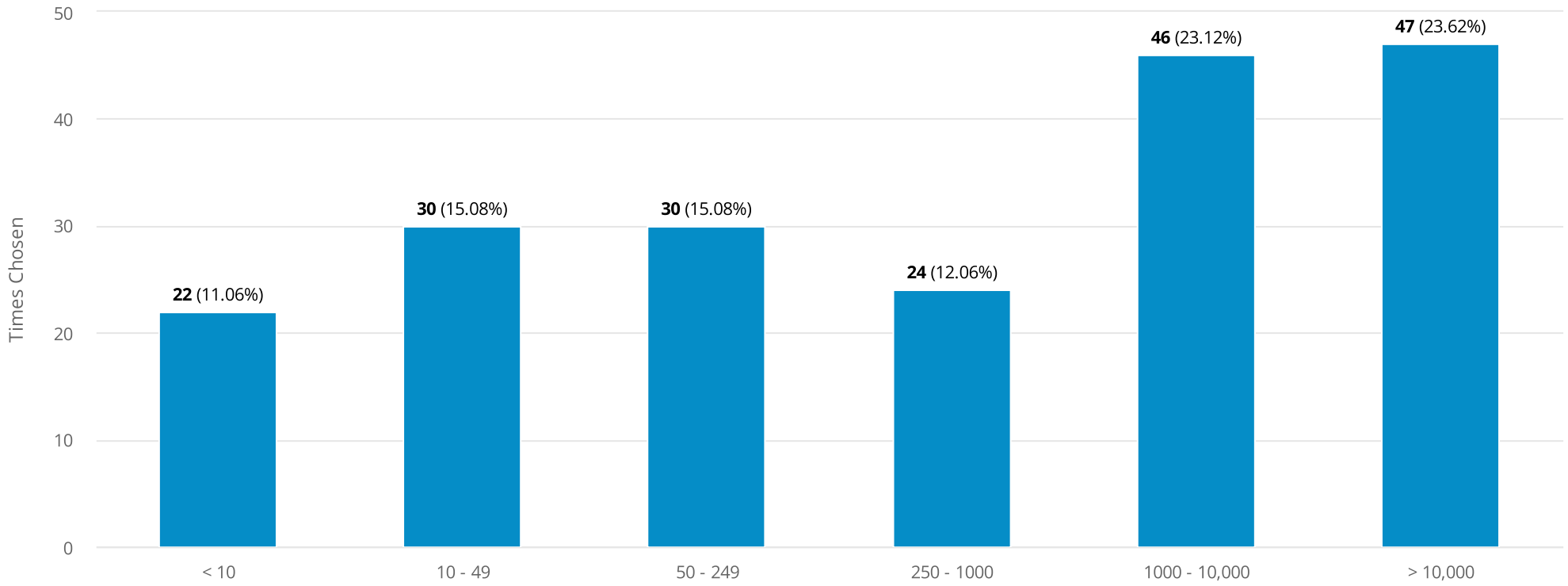
I work for a consultancy which works for companies in some of the above industries, primarily Automotive.

Motorized gates/doors

Other

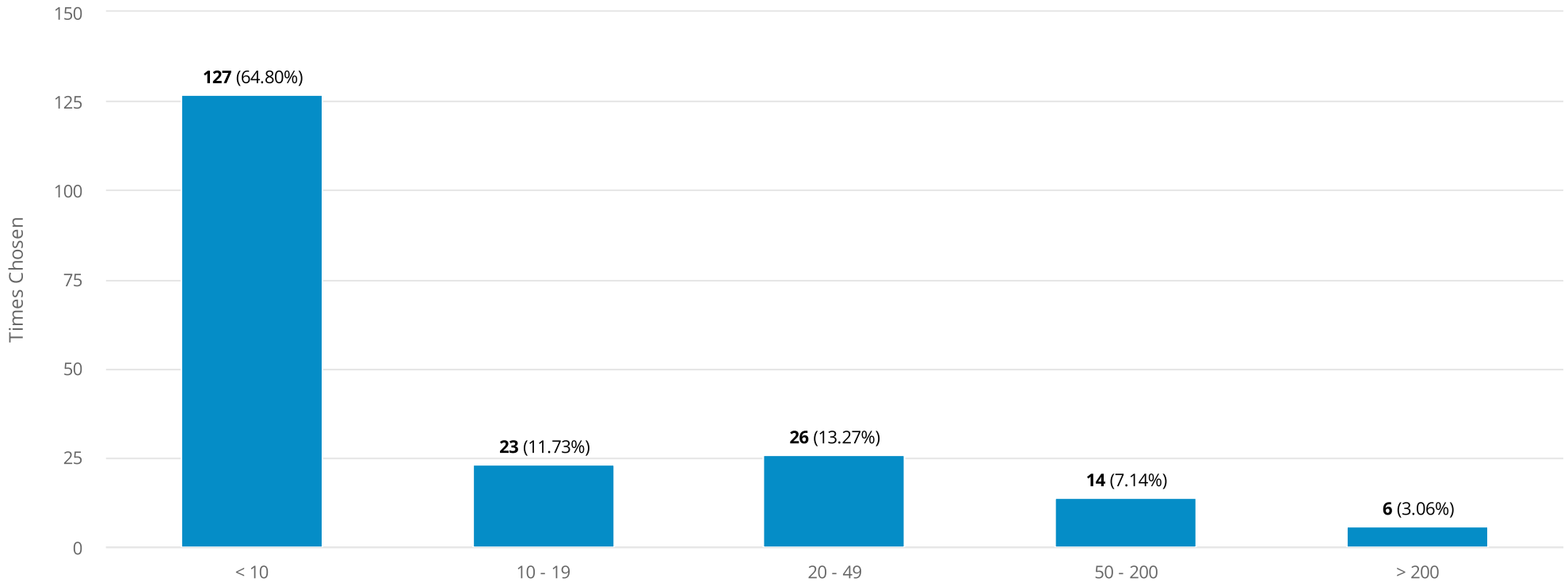
What is the size of the company that you work for?

Number of responses: 199



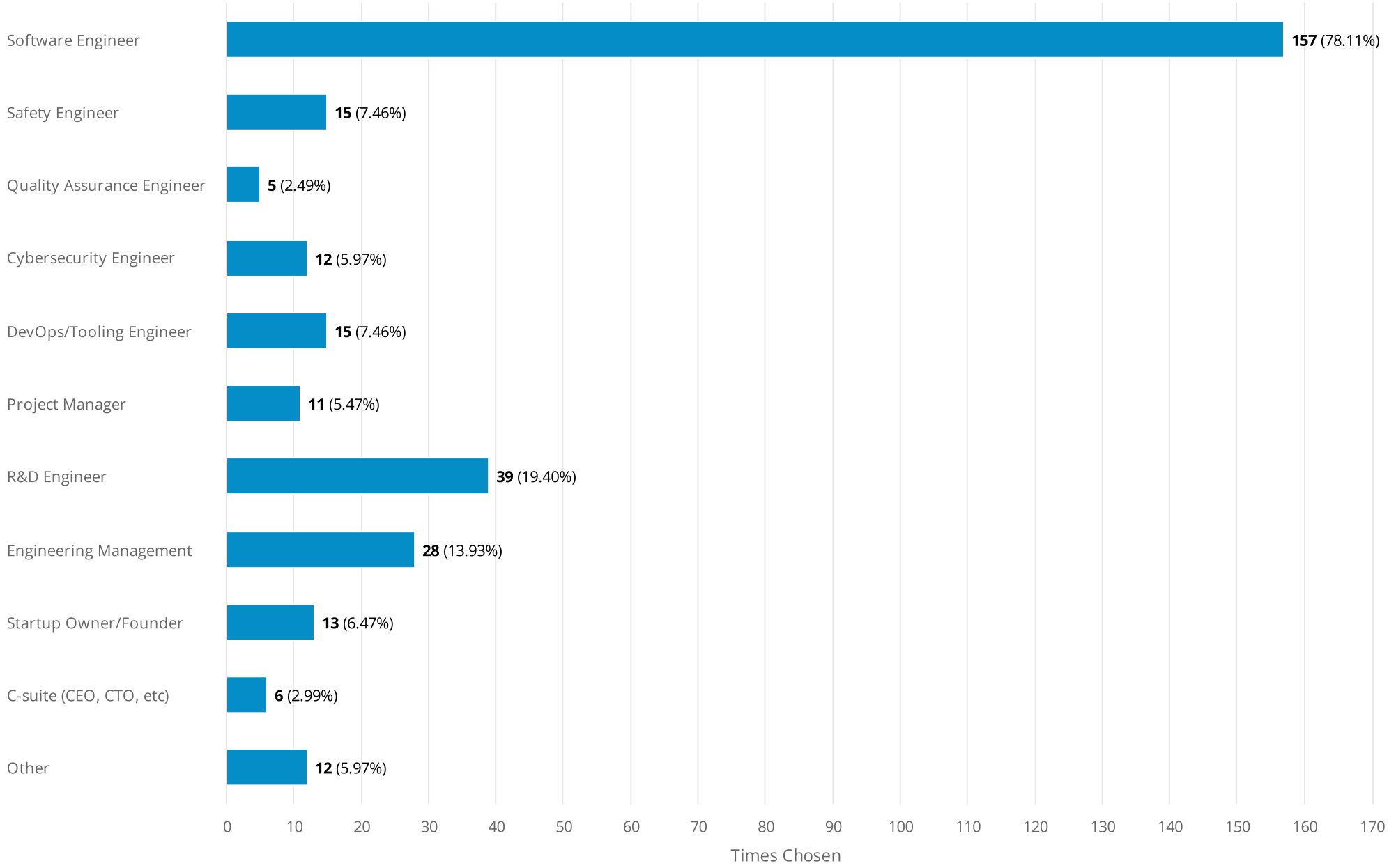
Approximately how many people actively write Rust at your company?

Number of responses: 196



What is your primary role or responsibility related to software development or safety within your organization?

Number of responses: 201



"Other" text answers:

Software Architect

Architect

Fullstack Developer

Roboticist

Product Manager

Senior engineer

SW Architect

Sales

Embedded EE

AIT

Product Manager

Other

Can you elaborate on your role?

Number of responses: 46

Text answers:

Working on a huge EU military aviation project

Chief Software Architect

Upper level firmware developer. Not able to independently make decision to use Rust, but has influence with the people who do.

Head of software and safety system architect.

I work on a software component that is integrated into projects using SoC's that have are used in vehicle ECU's that include safety critical functionality, and our component must be able to support safety concepts of the system (this generally involves being ASIL-D compliant by having 100% unit test MC/DC coverage, supporting freedom from interference and any tools used must be evaluated according to the ISO26262 process to determine the level of ASIL required).

Review code base as well as the hardwares before upload.

Software engineer working on perception for mobile robots.

Verification engineer

VP R&D

Writing algorithms for motion planning, control, state estimation and perception for robots

ADAS/AD SW engineer

Staff Engineer

Just a medior software engineer. Have used Rust for Embedded Systems.

I'm the primary developer of EtherCrab, a Rust implementation of the EtherCAT fieldbus protocol. I've done some consulting to help people implement it into their systems. Sorry to taint your data, but I feel it's important to give my perspective, even though I don't fit into the company/organisation role I think you're surveying for.

Embedded software engineer

TeamLead of a Software Team and representative of Helsing at the Safety Critical Consortium

Aerospace Engineer at a company that develops software for spacecraft.

Manage team of software engineers working on medical devices.

I'm the head of engineering capability for Software.

Ground software and flight software

Design & implement software for automotive controllers.
Define processes & develop tooling to support Rust in our processes.

Technical lead of a development team

Leading the SW & security strategy

Safety Manager and Technical Product Manager for the safety products.

Unfortunately I cannot elaborate much but myself and small team of enterprising "misfits" have been working closely with NASA and other entities to drive the adoption of Rust within our organization for use in Flight Software (human and other)

Principal software engineer responsible for all parts of the software development lifecycle process.

Integration and bugfix of sensor and parts in the production and new models.

Part Embedded & Part General SWE work

I am a principal software engineer working on embedded DO178C projects.

I'm a software engineer for 3d graphics in parking systems using primarily C++ with OpenGLES. I'm involved in training teams and an advocate for using Rust. However, it's not used yet.

Writing driver assistance software

Safety Critical Software Specialist

Functional safety engineer TUV certificate 10 years, process safety and machinery safety. Currently looking at formal safety for collision system in an iron ore stockyard. Last big project pyromet burner management systems for Albemarle lithium hydroxide plant.

I’m a senior embedded software engineer with 30+ yrs experience trying to guide new developers while still contributing to projects

I write Rust, mostly for performance reasons, not safety reasons.

Rust education, guidance, direction

PRA

Safety certification consultant with real project experience across many projects over the last 25 years around the globe.
Mostly RTCA DO-178 B/C DAL A through DAL A. cert RTOS development and Software Team lead at my last position for Airbus Zephyr project.

I am the Director of Vehicle Software at Stoke Space (launch company)

No

I'm a consultant specialising in software engineering process and tools development, with a focus on safety

I *lead* the team working on a safety related component, but I am not fully in any of the above ticked roles per-se.

I am the team lead and main architect for the project where we have introduced Rust.

Firmware engineer

Looking into software technologies required for automotive software development

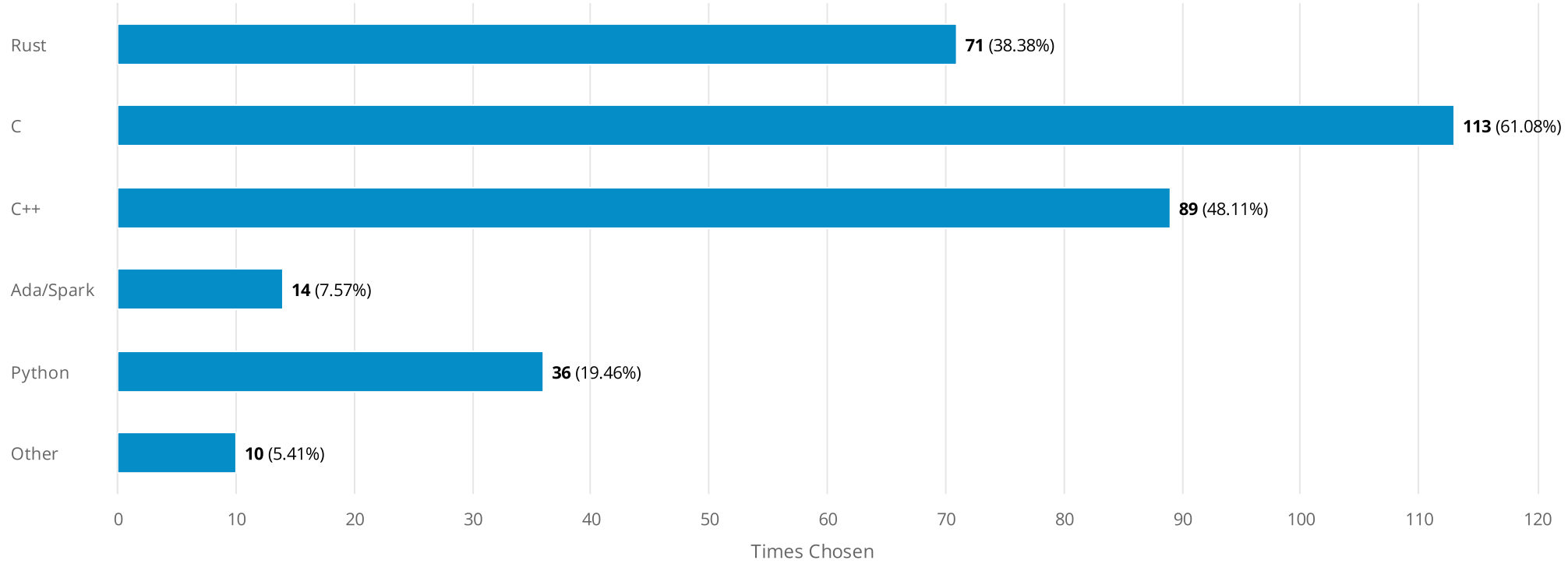
As the Avionics & Embedded System (AES) Team leader, I am responsible for:
- pushing the right tooling, ecosystems, SW Dev. env. and technology choices through team and projets
- supporting System eng. for safety-critical pack preparation, preparing req. specification accordingly and selecting processes & methodology according to the ECSS standards.
- supporting mission/operation SW life-cycle (i.e. ISS on-board SW deployment, cyber-security patching, update and network security)

I am a Rust ambassador, in touch with Ferrous System (Ferrocene) for bootstrapping the Rust approach in space SW...but still a long road to go I am afraid :/

#OpenForWork
benoit {dot} lietaer {at} gmail {dot} com ;0)

What language(s) do you use in your safety critical role?

Number of responses: 185



"Other" text answers:

Go
Mathworks Tools (especially related to MBD) :(
N/A - not exactly safety-critical for my current project; using Go

C#

Java, typescript

Object Pascal

Iec 61131 plc languages

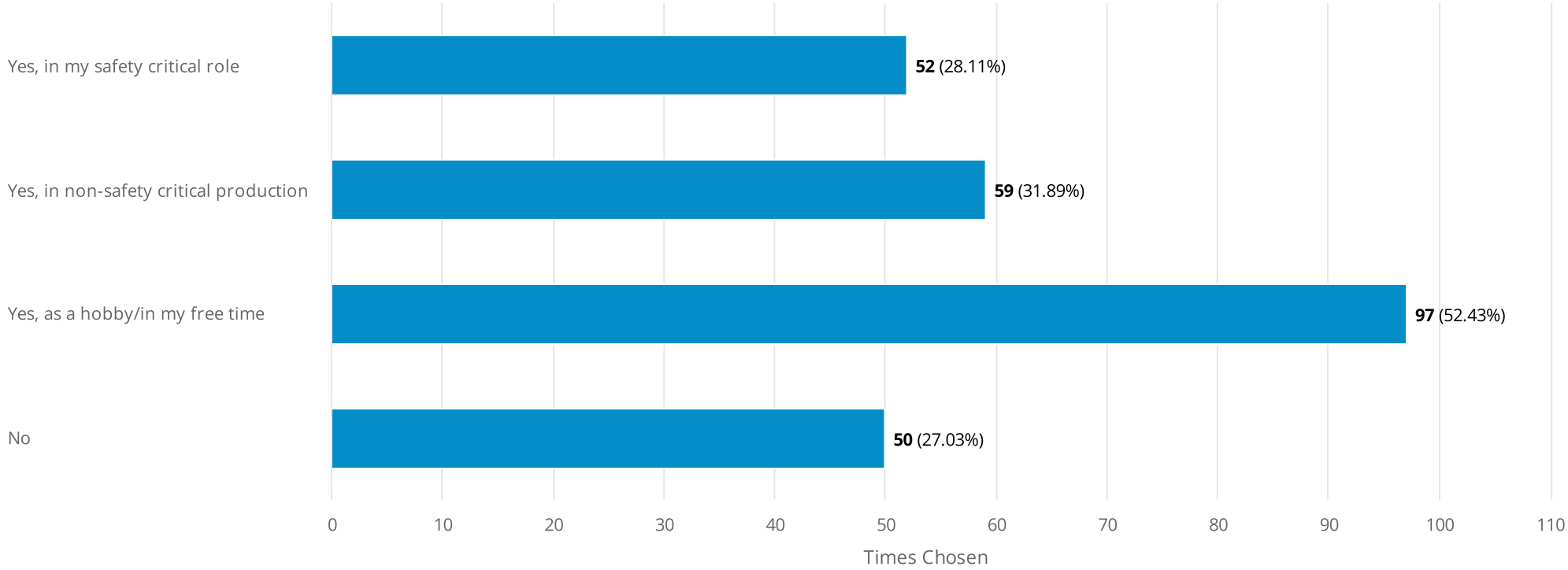
C# ASM

Bash

Java

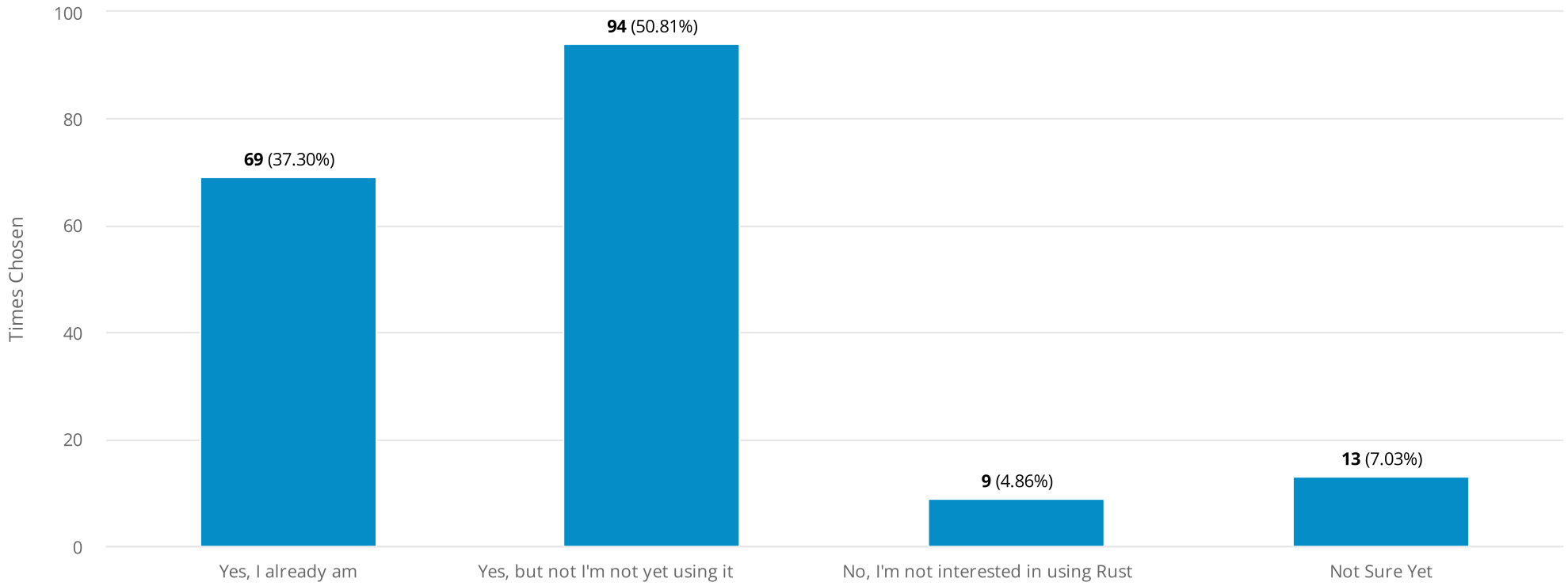
Do you currently use Rust?

Number of responses: 185



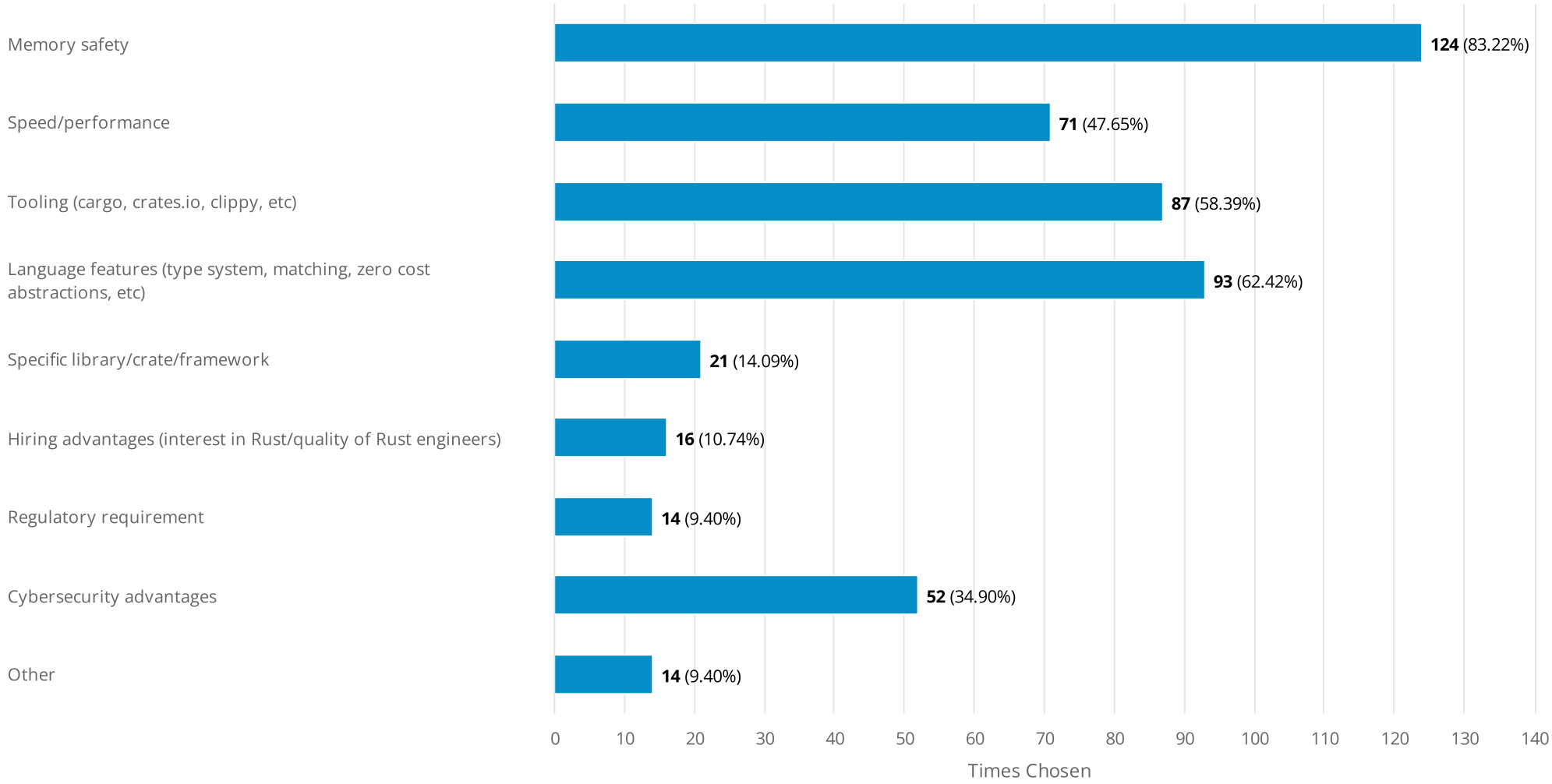
Would you be interested in using Rust in your safety-critical role?

Number of responses: 185



What were your/your company's primary reasons to switch to Rust?

Number of responses: 149



"Other" text answers:

Just testing the waters, some devs liked it

nobody dares trying, yet.

No one care about it at the moment :(

Technology stack standardization

Cost efficiency and it is just fun programming in Rust

The company had no interest in Rust, some engineers just started using it (without asking) for non-approval relevant SW because they liked it

Productivity

Did not

Not using Rust

Client companies interested in Rust

none

Doesn't switch to Rust

Development ease

paving the road-to-space standards (ECSS), should be a no-brainer...still a long way to go :/

What are some of the advantages to Rust in your role?

Number of responses: 29

Text answers:

Mostly for safety, tools around it. Simplicity of code compared to C++

- Rust is inherently safer than C/C++ which is the alternative
- My developers find it super intuitive

Everything that is easy to get wrong in c++ is hard to get wrong in Rust. We have a ton of engineers who only know c, and to train them to use modern c++ "properly" when you have so many rules that are hard to explain on the outset (e.g. "rule of 0/3/5", or how to explain when the compiler will start using virtual function pointers for polyphormism). Rust just makes much more sense.

Getting unexperienced programmers (electronics engineers, control engineers, etc.) up to speed in Rust is much much faster. They are confidently making stuff in Rust in a few weeks. In C++, this took ages.

C++ is widely used, and Rust would be essentially the better replacement for it from a pure language perspective. I'm using Rust for some hobby open source projects, and some things I love:

- modern syntax with sane defaults vs. C++ (e.g. requiring explicit mut)
- enums + macros (e.g. serde) that are really nice to use (PITA in C++)
- strict compiler + actually useful compiler messages
- great documentation and a set of established standards

- tooling in general
- community

Control and performance of C++ with better guarantees and less legacy problems, better build systems

Asynchronous programming with Rust.

For development, all the ergonomics of the language and tooling. For the end application, performance and explicit thread safety. Existing C EtherCAT implementations just kinda go "yeah it's thread safe, trust me bro" in their docs. Having this explicitly guaranteed by the Rust compiler lets application developers go crazy and make better software with fewer wait points for overly cautious mutex guards, etc.

More reliability and trust in the code, easy cross compilation, etc..

"Undefined behaviour" is treated as a bug, not a feature.

Nice tradeoff between usability and various notions of safety.

Many additional restrictions in C/C++ like MISRA-C are not necessary (or better integrated in the compiler/clippy if they are).
Lot less effort regarding freedom from interference.

Many issues which we have in C++ are prevented in Rust during compile time.
The language and the compiler saves a lot of analysis effort which is performed manually for C++.
I think we can save a lot of maintenance and debugging effort.
Developers feel more confident if the compiler is supporting them.
Besides ownership Rust can also support in deadlock prevention.
Rust is more explicit then C++, which makes it easier to understand whats actually happening under the hood. In C++ this was a problem for many developers.

Writing high-quality software for less money.

Type safety, Clippy, borrow checker

In my specific role of baremetal embedded, toolings and languages did not evolve since pretty much C.
Is not uncommon to see company without proper automated testing, pipelines, often because proprietary build system/compiler incompatible.
Much more powerful macro system

- relatively hassle free compared to C++ projects

The way rust handles enums is very interesting. It pushes you to define all possible values, and then ensures that you handle all the values. I've seen this make engineers really consider how their software will handle failure conditions/states.

Rust is designed for safety, and thus requires less external tooling like MISRA checkers. Tooling is excellent, especially clippy, but also cargo and semver checking.

Not using Rust

Cargo, Performance, WebGPU tooling, WASM

if it compiles, it runs

Interested in the adoption of RUST within Aerospace projects, but I have my reservations at the moment

Development environment is huge (rust analyzer is incredible), tooling like Tracy for live profiling, generally if it compiles it runs.

It should be all about Cybersecurity

Familiarity, memory safety, typestate programming, static checking of correctness. Things like `cargo deny` and `cargo outdated` along with clippy et al.

The fact that one can so effortlessly write code that runs just as well on a bare metal target as on a host pc. This means that you can test and verify your code much more easily, but also that one can re-use the code modules indented for production as sub-modules in various host based tooling that needs to perform a similar function. A simple example of this is a parser. You can write it once and then use it pretty much everywhere.

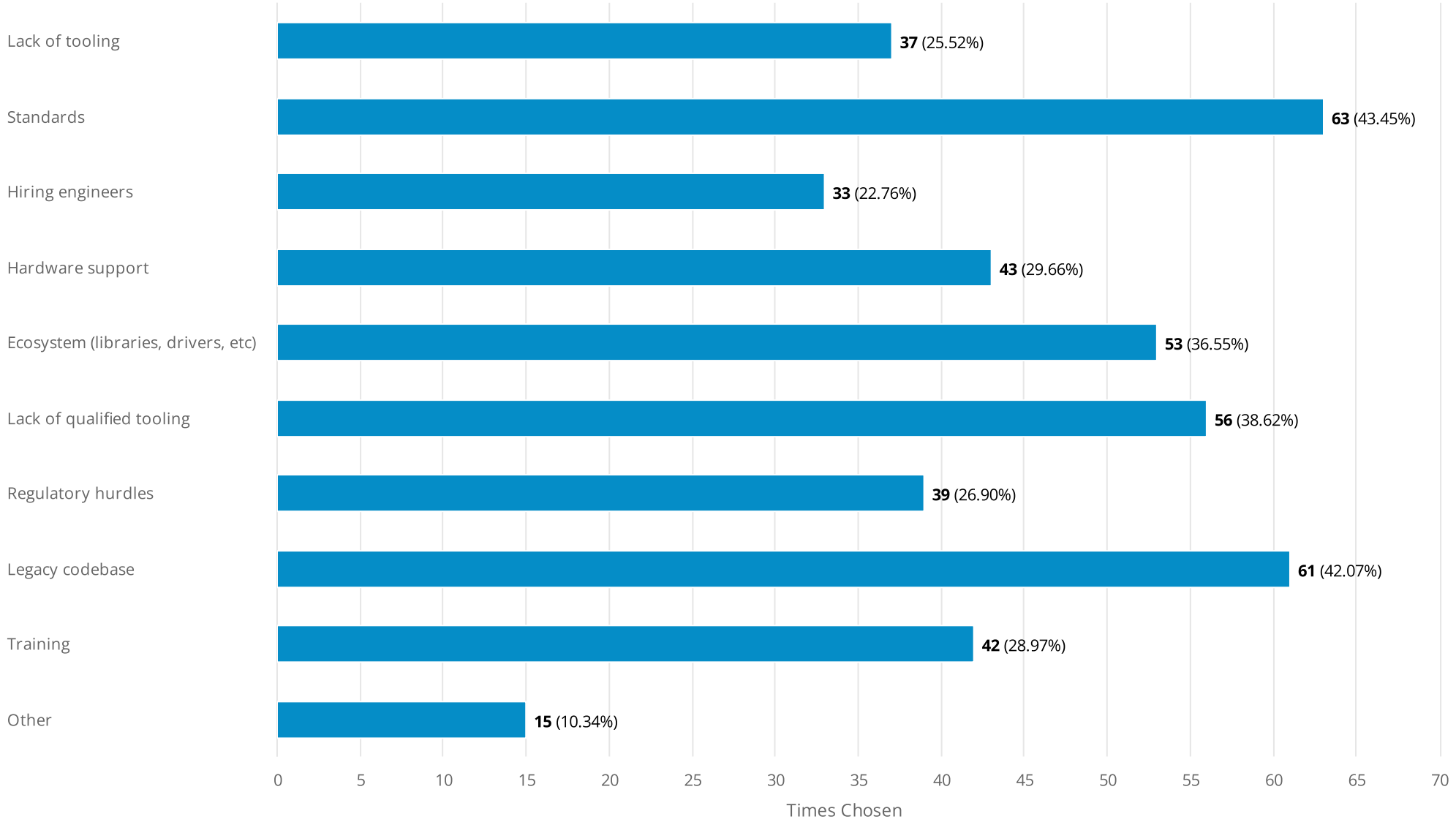
Safety critical Rust is much more similar to everyday Rust than safety critical C++ to everyday C++.
The painful debugging on hardware is much less necessary with Rust.

Superior tooling and build system when compared to C/C++

- embedded ecosystem is just a dream come true, with probe-rs, dfmt/rtt, ..
- PAC/HAL 'standardization'
- cargo + crate is the best SDE ever used
- safety-critical WITHIN the compiler should be a ticket to space...yet I struggle forcing it in project, just because it is not stamped 'ECSS compatible' (Yes, ESA is _that_ simple).

What are the primary blockers or disadvantages to using Rust in your safety critical role?

Number of responses: 145



"Other" text answers:

easy interop with existing C++ libraries would be awesome, but I know it's hard...

we are only providing software libraries to our customers. Our customer do not use Rust.

Lack of projects that require Rust usage

Corporate IT cybersecurity policies blocking access to crates.io

Does not exist as long as known tools. Trust need to be build.

Customers need to use multiple compilers and support multi language code base

current team does not know rust

momentum

debugging rust on an embedded platform is challenging

Owners want to draw down on standard plc languages and programmers , eg chevron, bhp, woodside, etc are not interested in owning anything not standard industrial control gear

Fear of change

organizing build process in the org

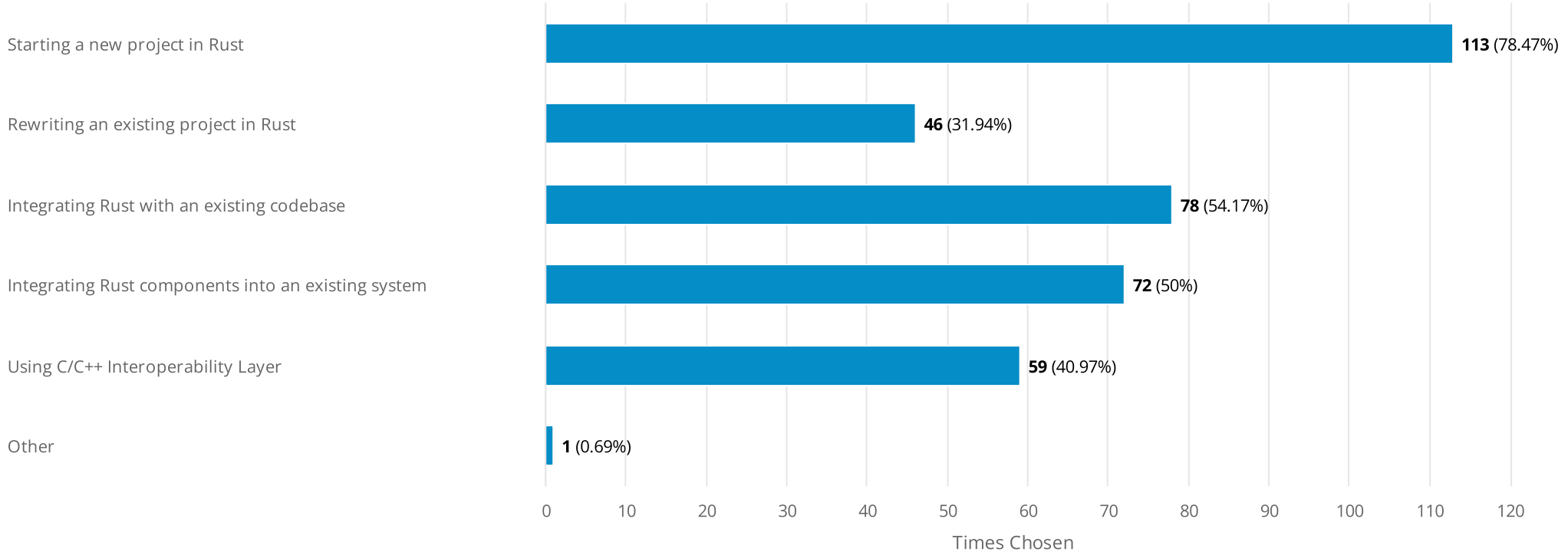
Political Inertia

Lack of well established auditor relationships.

Safety-critical WITHIN the compiler should be a ticket to space...yet I struggle forcing it in project, just because it is not stamped 'ECSS compatible' (Yes, ESA is _that_ simple).

How do you or would you be interested in using Rust in your safety-critical role?

Number of responses: 144

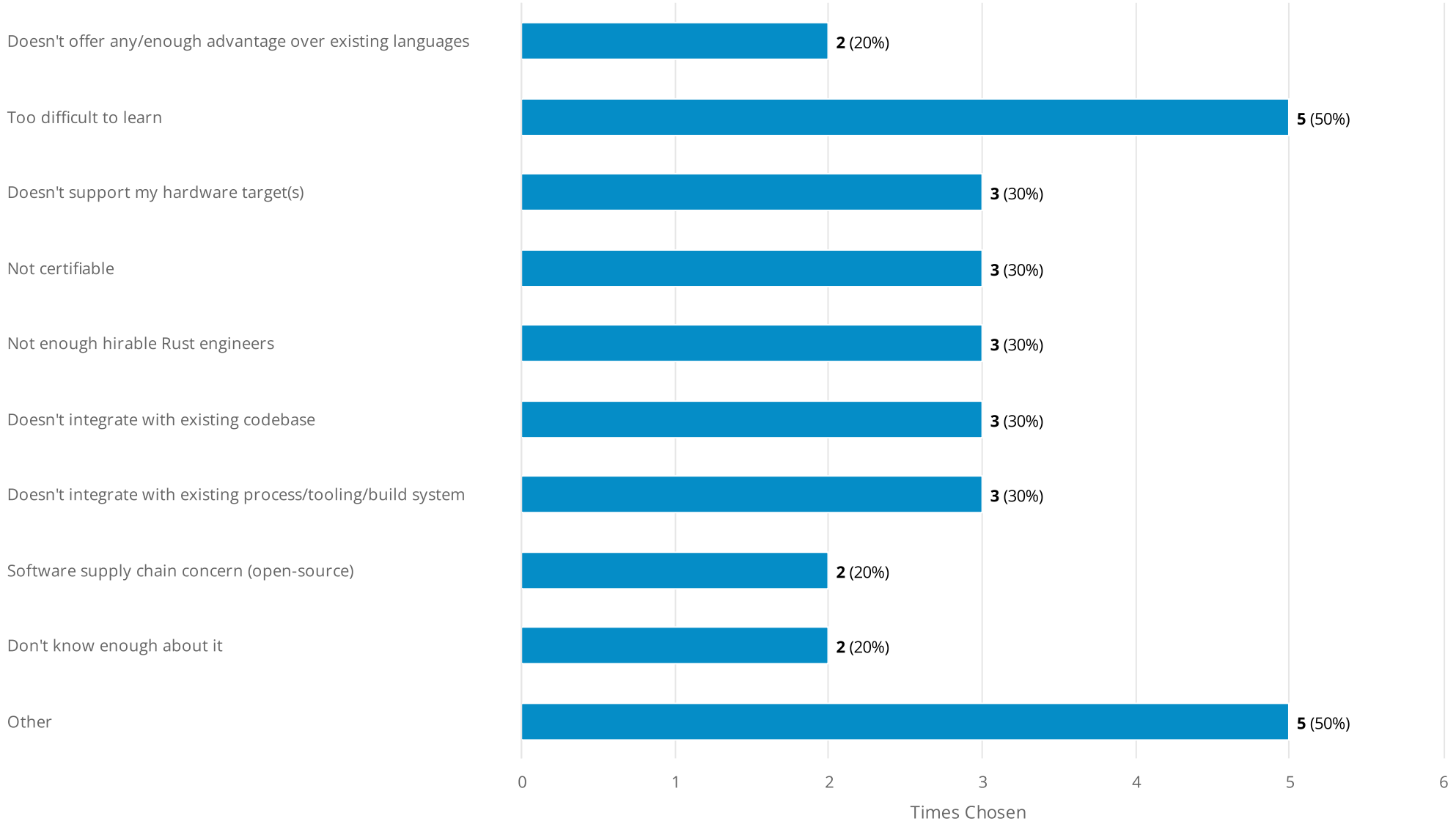


"Other" text answers:

Client asks or a specific need emerged

What are the primary disadvantages to using Rust in your safety-critical role?

Number of responses: 10



"Other" text answers:

Rust looks boring

slow builds

Its shit, maintaining it sucks, its hard to read

They lie too much about their safety guarantees

Cost to update existing codebase, reskill engineers

Can you elaborate on the downsides to Rust for your role?

Number of responses: 2

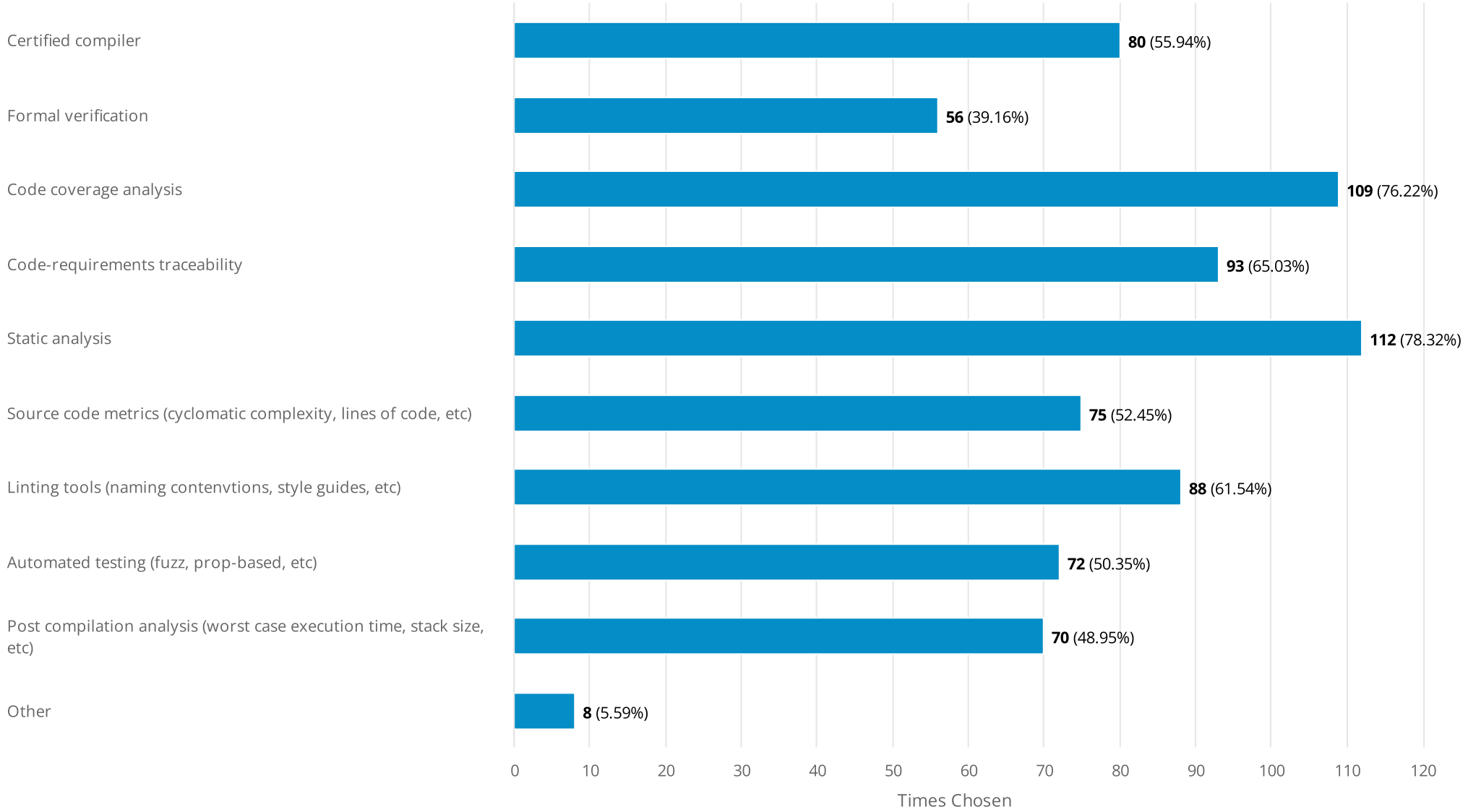
Text answers:

Rust is fine per se, esp the tooling and the external libs. But it's unsafe per default, and the syntax is horrible. There are better, more readable and actually safe system languages out there

Simulink. We use A LOT of Simulink. Controls engineers didn't learn C, they're not going to learn Rust. If you could get Mathworks to add a Rust target things would be amazing.

What types of safety critical code tools does your work require?

Number of responses: 143



"Other" text answers:

Tests, lints, and type system that can be trusted. All already part of Rust. :)

Using languages with an internationally recognized standard (e.g. ISO)

N/A right now

Accept from functional analysts

Model based development and code generation tools

It depends on the required sil level, as per techniques and measures in 61508

Code Object code mapping (DO-178)

For (ESA/EU) space domain, we need _now_ formal ISVV, coverage, ECSS certified compiler and

Are there "best-in-class" libraries or solutions in another language which have features lacking within the Rust ecosystem for your use cases?

Number of responses: 30

Text answers:

Green hills Ada compiler for powerpc - 32bit
LDRA toolsuite for testing

Tools exist, the certified tools are the problem, for example, Rust compilers

- Mathematics (CMSIS-DSP) - there are Rust crates but none certified
- Hardware abstraction layer.

1. Memory Separation
For safety critical systems, it is common to separate the "safety critical" memory from the "non safety critical" memory. Then the MPU is used together with the RTOS such that safety critical components are run in a "priveleged" mode such that it can access the safety critical memory, but other components that do not need to be developed according to ASIL-D level are run in a lower mode such that the MPU blocks it from accessing that memory. In order to help enable this, AUTOSAR has developed a standard way of specifying memory sections for variables of components. The linker file is then used to place the various sections into memory regions that the MPU will setup accordingly. I have not really looked into it, but I am not sure how trivial this would be in Rust.

2. Default of switch case
In safety critical software, you **must** be 100% sure what your software will do, even if memory gets corrupted (i.e. you must be defensive). For this reason, MISRA specifies that a switch case shall always have a default case, even if you are switching on an enum and you covered all "values" of the enum.
In Rust, the "match" does not allow you to have a default. So I am unable to say if someone puts a bad value in for the value stored for the enum (or memory gets corrupted as it enters my function), I do not know which path my match would take.

not really, but I'm also not working on strict safety stuff

ROS library support
Numerical libraries for optimisation, dynamics, etc.

in general: code coverage measurements for embedded.

Not really from what we've found

SystemWeaver with language integration or similar, easy to use/customizable static linters

This is less of a library and more of a feature. But refinement types are super useful in safety critical software. Rust is also close to having good unit support, but const generic expression aren’t stable yet

Tooling for post-compilation analysis are still lacking. For example, "-Z emit-stack-sizes" is nightly only, and there are no plans to stabilize (https://github.com/rust-lang/rust/issues/54192)

This is a very basic analysis tool that's missing, for example.

GUI, better IDE support, a .Net style framework for enterprise applications

Eigen C++ numerical library

N/A

Code Coverage: VectorCAST
Static Code Analysis: Bauhaus
Code Metrics: Self-made
Post Compilation Analysis: Self-made, valgrind, ...

VectorCAST

Embedded Template Library (statically allocated standard lib replacement).

Targetable safety standards like MISRA, along with static analyzers that support it.

Simulink modeling and C code generation

visualization, plotting libraries are a weakness in the Rust ecosystem as well as any computer vision libraries (traditional or other)

No

Code readability, Rust doesn't have Ada-like syntax. Official ISO standard is also missing.

GUIs

There are best in class solutions available for Ada and C, C++ is catching up, but it's a C subset with classes and limited functionality

I don't think so. Maybe if rust had better ROS 2 support

not yet

Operating System Standards (Autosar, Zephyr)

vsomeip (SommR would have been a solution)

Linux interfaces (CAN, raw Ethernet, etc...)
Distributed embedded messaging

Simulink.

Eureopean Space Agency is still blind to anything that is not stamped 'MISRA': showing/prooving that this IS part of Rust would help...but I am afraid this can only be done by sticking to a (ECSS) standard/certification.
Having more 'space' grade HW component (MCUs) libs would be a nice-to-have...but I am ready to write it (remember we use very old, dusty but with mission-heritage devices, yes, we still run SPARC/LEON MCUs,....praying for RISC-V to fly).

Are there any libraries that are not written in Rust that you would like to use in the Rust ecosystem?

Number of responses: 23

Text answers:

Yes

Windows GUI libraries (WPF or similar)

not safety critical, but for robotics:

Eigen
OpenCV
VTK
Ceres
PCL (or rather an equivalent, because PCL's API is terrible)

in general: code coverage measurements for embedded.

KLEE symbolic execution/verification

CGAL (unrelated to my industrial-focused response in this survey, but I do a lot of geometry in my day job which CGAL would be useful for)

Qualcomm fastADAS, SOME/IP, CAN

NASA cFS and Fprime could use better rust support

Point Cloud Library (PCL)

Probably, but nothing (yet) comes to mind :)

An enterprise-ready framework like .Net or Spring Boot to decrease the learning curve and development time

N/A

ROS2 - there are partial rust versions and some require linking against c++ version

OpenGLES

No

Embedded peripherals need a lot of help. I2c slave implementations, etc.

GUIs

N/A

No

Zephyr

basically any AUTOSAR specific APIs

Not

- ECSS certificed compiler
- insisting on tooling equivalence, such as vectorcast...

Are there any crates in the Rust ecosystem that you would use in safety-critical software if it were certified?

Number of responses: 32

Text answers:

cc, rayon, tokio

Async libs, No std crates

Certified WASM Runtime

heapless

- Heapless
- libcore
- rustfft

Yes

embassy
serde
thiserror
anyhow
hyper
tower

RTIC, heapless, serde, postcard, all of cortex-m

Maybe ndarray

Nalgebra

Many. But most "certified" anythings are automotive and industrial, not aerospace. That's what we're trying to fix. :)

I would use crates that are well maintained and has an acceptable license.

cargo-llvm-cov, contracts

N/A

- Argument parsing
- Parser framework
- Async runtime
- Lots of small things e.g. about byte ordering, synchronization primitives, hashing, etc.
- Logging / tracing
- Containers that really support strict allocation handling
- Serialization framework

Many...

serde

Deku, unfortunately it uses alloc under the hood. An equivalent declarative “serialize/deserialize” for arbitrary binary protocols without alloc would make Rust an obvious winner for spaceflight and defense network protocols.

opencv-rust
nalgebra is awesome but is it certified??
crossbeam_channels ?
ROS2 - Rust certified?
DDS ?

Tokio, Prost, Zenoh

No certification needed, but ST's HAL and embassy-rs or rtic are all we would need.

serde, cbor, log, smol,

num-quaternion

Cryptographic implementations

Only if the client requested such a path

Verification of no unsafeties, ie no unsafe blocks, no numeric overflows, no dead locks.

Not Sure

Nalgebra, ros2_rust, and many many more

rtic, heapless

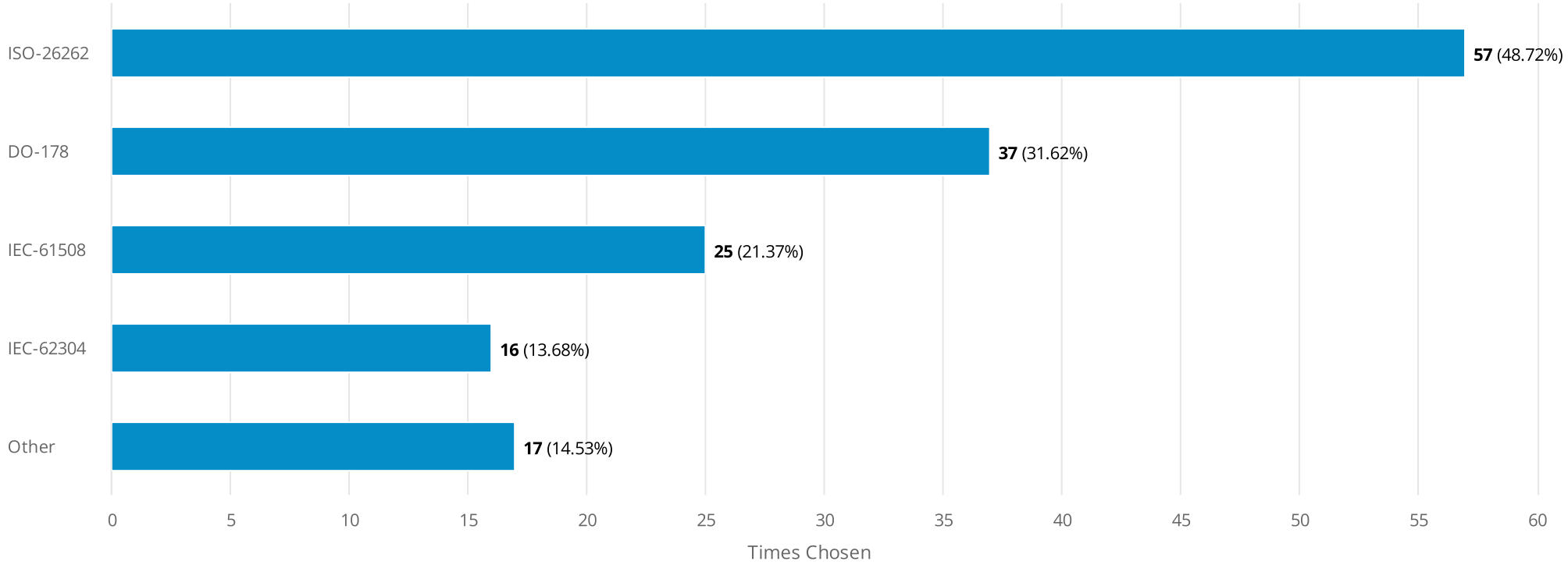
chrono, num-traits, heapless

We don't have certificate requirements

Yes... pretty much all of it ! :)

What standard(s) do you work with in your safety critical work?

Number of responses: 117



"Other" text answers:

EN 50126, EN 50128, EN 50129

IEC-50128

ISO-25119

n/a

MSC.1/Circ.1512

US DoD internal standards

ISO-13849

FDA Guidance on Cybersecurity, CISA 2026 Memory Safety guideline, CRA 2026

NASA Computer Based Control Systems and Class A software standards

ISO13849

EN 60079-29-1

61511 and 62061

DO-278

PKCS#11

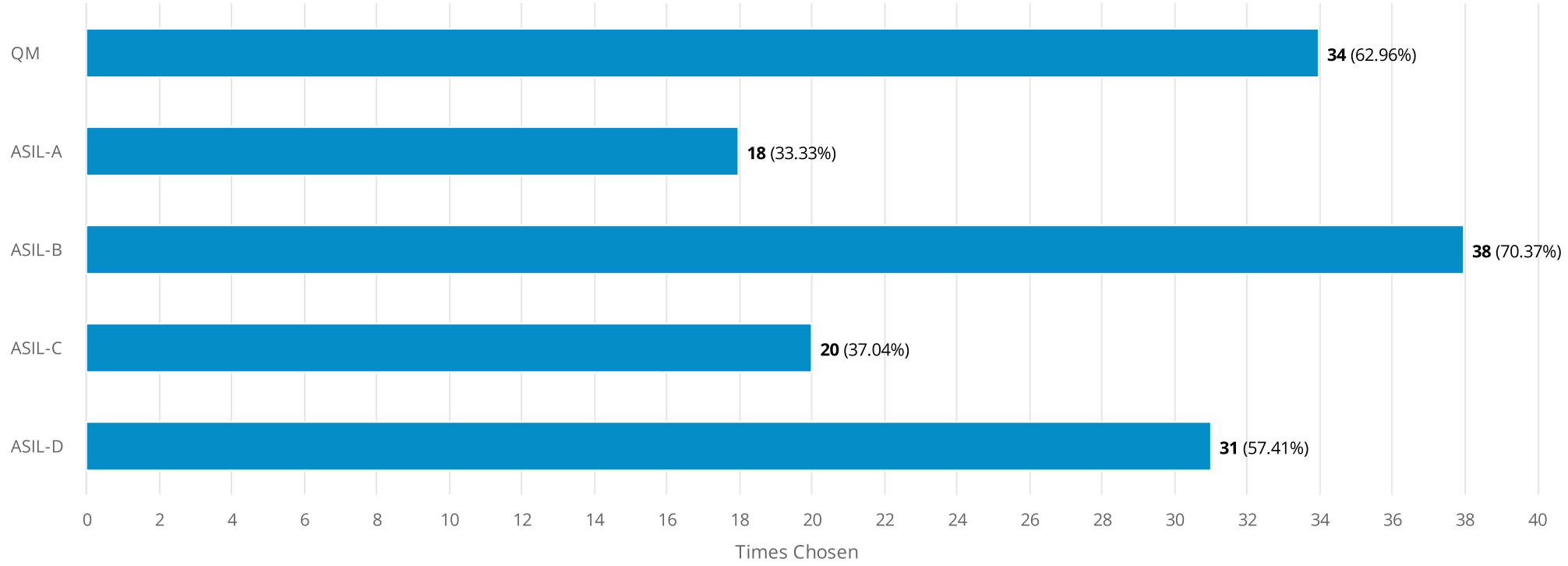
Range 91-710

Industry-specific life safety requirements (not ISO/IEC standard)

ECSS

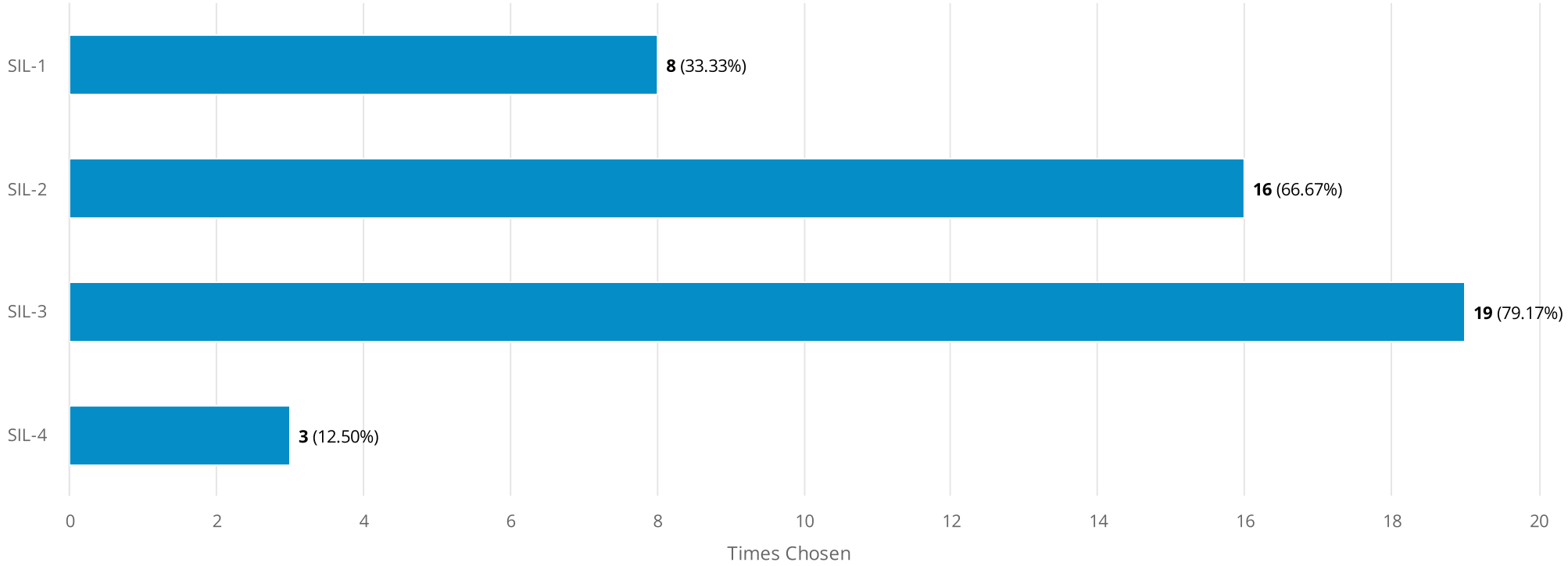
What levels of ISO-26262 do you work with?

Number of responses: 54



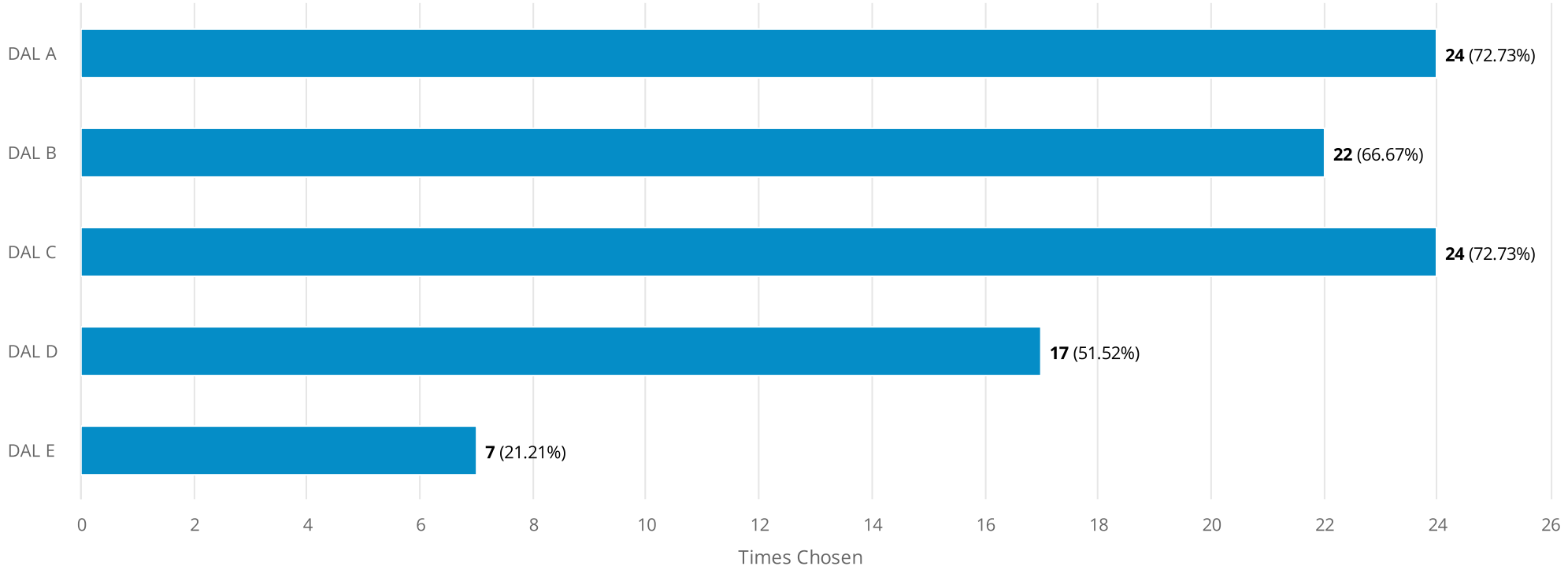
What levels of IEC-61508 do you work with?

Number of responses: 24



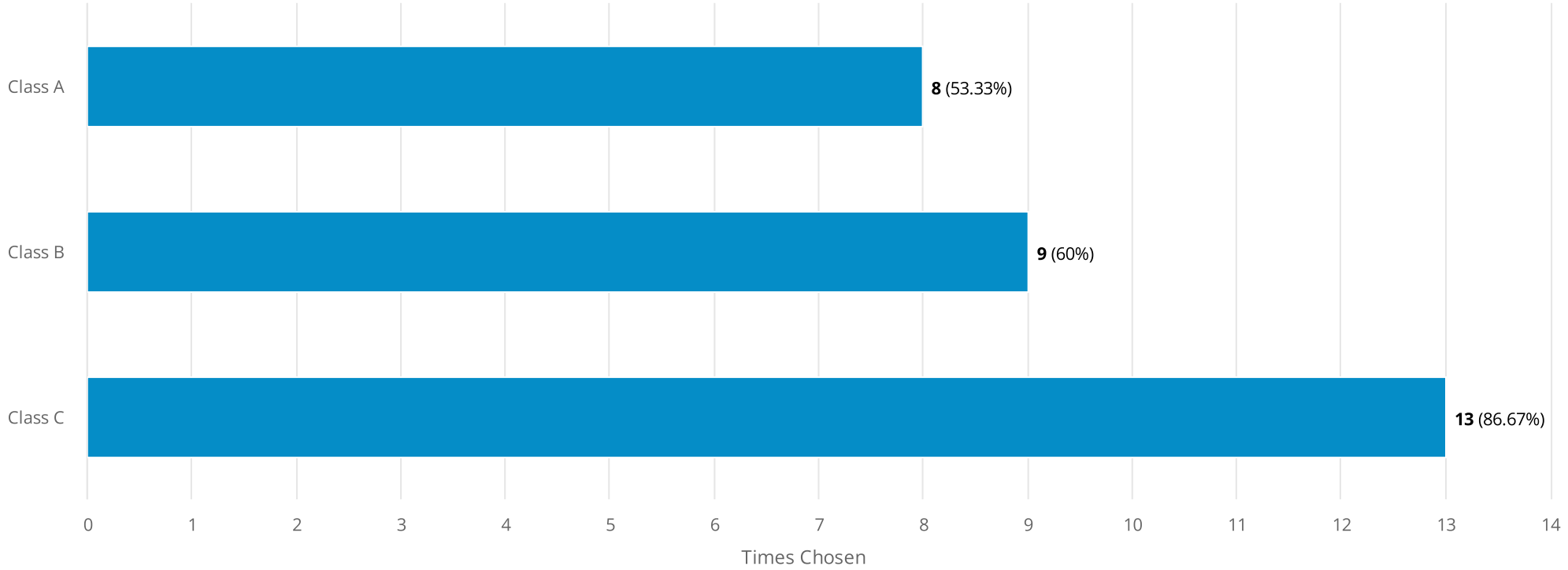
What levels of DO-178 do you work with?

Number of responses: 33



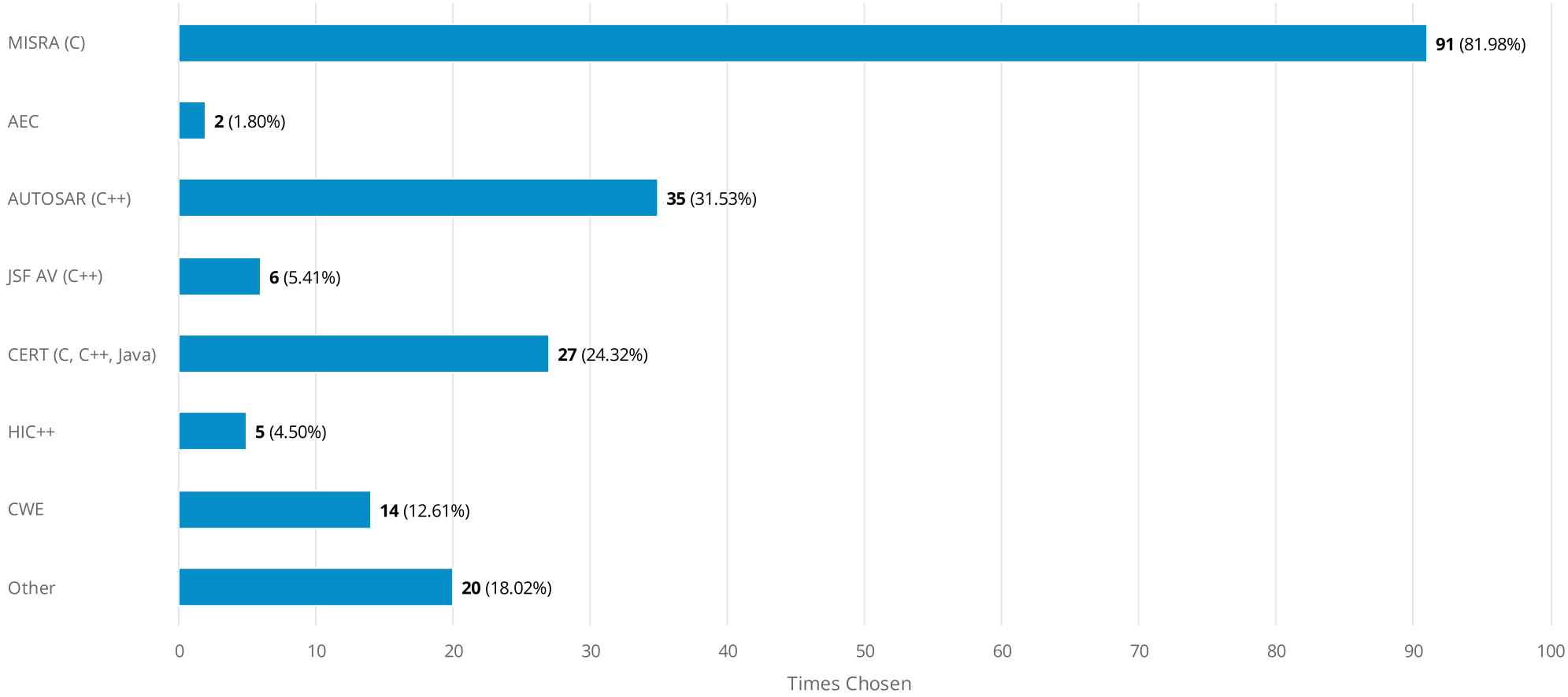
What levels of IEC-62304 do you work with?

Number of responses: 15



What coding guidelines do you work with in your safety critical work?

Number of responses: 111



"Other" text answers:

We need to define our own.

Still under investigation

MISRA(C++)
n/a
Other
SPARK
Other
Our own coding guidelines, based on readability
Our own based on the above
Internal guidelines based on MISRA/cert in c++ land, a less intensive standard based on ANSSI Rust Guidelines for our existing Rust
MISRA C++ 2023
In house SPARK Ada coding Standards
Iec 61131
In-house.

Project created DO-178C coding standard

MISRA C++

ANSSI

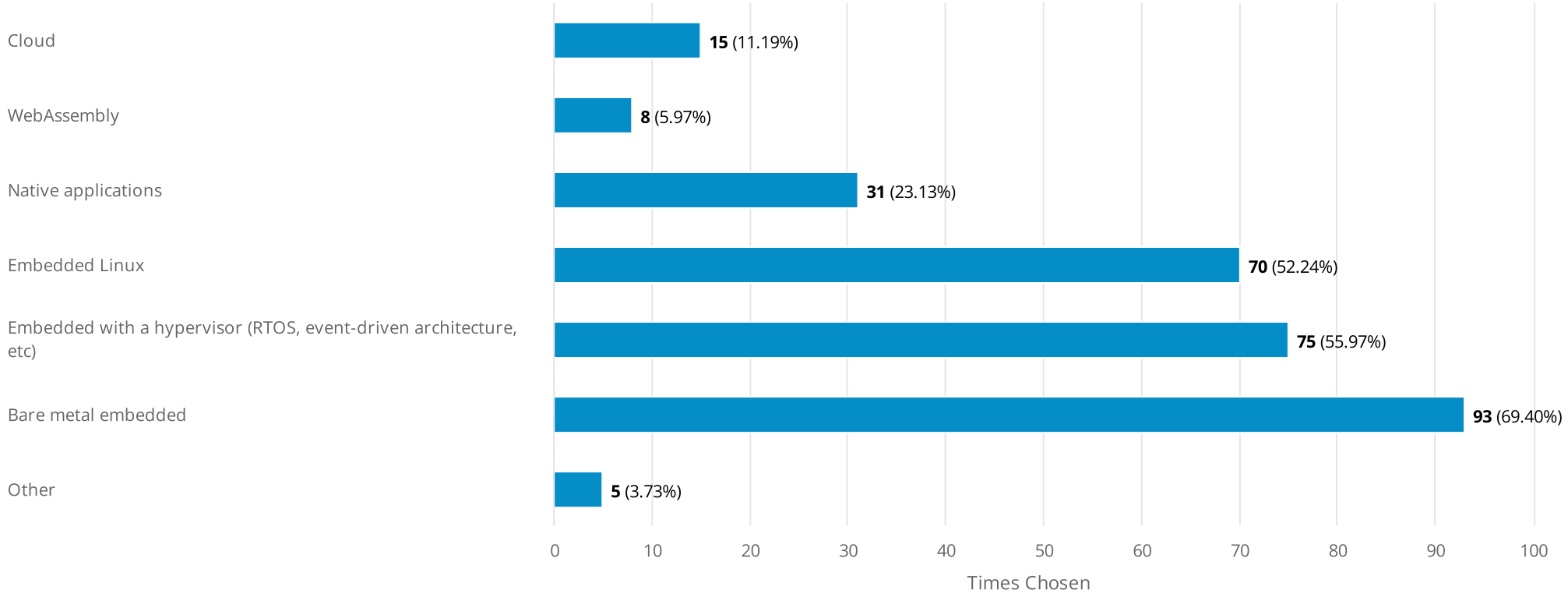
company made

C++ core guidelines

MISRA C++:2023

To what environments do you deploy your safety critical code?

Number of responses: 134



"Other" text answers:

Our hard safety critical stuff runs on a controller that is separate from the high-level software.

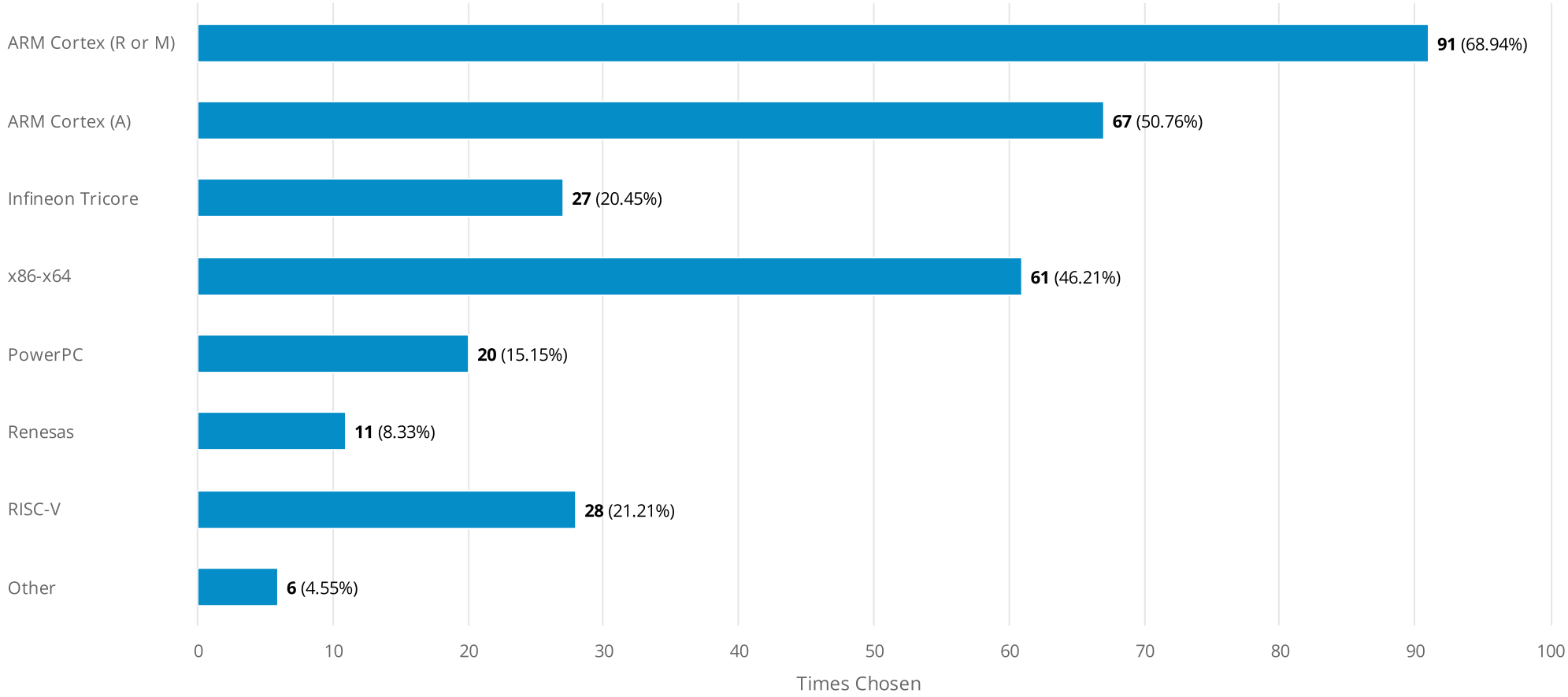
QNX,...

qnx

DAL A embedded commercial of the shelf operating systems
Industrial logic controllers, eg hima. Siemens etc

What chip architectures do you work with?

Number of responses: 132



"Other" text answers:

MSP430, SPARC
LEON

It is abstracted from me, certification (eg by exida or tuv) is most important

MIPS

MIPS, PIC

SPARC

What kind of support or resources from the Rust community or the Safety-Critical Rust Consortium would be most helpful in enabling Rust adoption in your industry?

Number of responses: 48

Text answers:

Hardware support.
Documentation should be more elaborate on how to compile to baremetal system and work with assembly code.
Some Math Library that precision level we can select e.g. sqrt how many levels of precision we want that also to be covered.
All popular hardware should cover for baremetal compilation.

We really need a certified Rust compiler for DO-178 DAL-D / DAL-C ASAP! This issue is currently the most critical one so Rust adoption can be performed in our company. If this does not happen in a foreseeable time I assume Rust is dead at our company.

Mainly safety standards to qualify the Rust code to ASIL standards

- coding guidelines
- tooling for code coverage
- streamlining process for certifying/assessing 3rdparty crates
- ecosystem for certified crates

We basically need a MISRA kind of "authority" to come and say, "if you follow these guidelines, you're following best practice". I believe MISRA has already made some strides towards a MISRA Rust guide but I believe nothing official yet.

Introducing Rust at more universities, also outside of computer science majors.

Drive robotics adoption by building libraries, enhancing ROS interoperability, and even training.

qualified compiler
certified most common crates

Formally standardizing it through ISO

TBH, I think a big hurdle is that many devs haven't used Rust seriously yet to know its advantages. Maybe hearing more about "success stories" could help.

62304 compliance

"Safety critical" features aren't necessarily the most missing features in my industries, it's a numerical ecosystem that's holding back complex cyberphysical applications. Numerical libraries need to be highly robust and mature, and are mostly written by PhDs in the topics. Robotics people rely on those tools and can't necessarily implement them themselves to a robust level.

No dependencies in any external libraries, or an affordable, certified Standard lib.

Improved support on non-cargo build systems (e.g. Bazel) for mixed language builds. Development of industry-wide coding guidelines for safety critical applications. Development of tools to support those coding guidelines and ISO 26262 activities.

Understanding how to leverage Rust to make processes simpler and faster, right now we have to figure it out ourselves

More advocacy resources to convince customers (OEM) to switch?

BSP Support, QNX Support

A set of coding guidelines (as currently in progress), tools for requirement tracing, guidelines for easy creation of custom static lints

Having a DO-178C certified compiler would go a long way

Evangelizing certified toolchains and getting an ASIL subset of the standard library.

Possibly some coding guidelines too

The Rust community has been fantastic!

Guidelines, examples, business/management-level training about why memory-safety is important

Safety critical rust coding standards and certified tools/libraries.

Easy reliable way of switching. Migration from C++ to Rust.

Possibility to fulfill ISO26262 e.g. regarding coverage (MCDC)
Certified commonality crates for ASIL usecases!

Hosting a service like crates.io company internally

Tool chain availability, documentation and support from vendor for aurix tricore, drivers availability, MATLAB simulink support

Any collateral related to safety critical aspects would be hugely helpful at this stage as we are actively trying to put together guidelines.
coding standards, best practices, tools, ABI stability, language specification ,

BSP support for bare metal.

dynamic + static analysis

Frictionless C/C++ developer training, possibly pool of good developer, maybe trough certification

- good RTOS options (including tooling and board abstractions) like Zephyr for embedded C/C++ applications
- knowledge of approval agencies accepting it's use

Having a set of crates that will always work together would be ideal. We have seen issues where a crate that is a dependency of another crate has updated and broken our build pipeline. Ideally I'd want to know what has changed within the crates that we are including in a project, and I'd want to be able to control when they change.

press release/article about a successful project of a certified software component, that shows using rust is a viable option

Help to reduce the amount of paper work that needs to be done to justify that Rust can be used for an automotive project. Maybe some publicly available document or knowledge base which helps to get started with Rust. As a software engineer (with some limited knowledge about safety standards and the qualification processes) it would be good to have some kind of instructions or checklist, etc. on how to get Rust into a safety critical application.

Stable ABI

create of a high integrity development standard

Convince clients/owners

The chip I work with is not supported.

More aerospace involvement

I guess defining a subset of RUST that is safety certificable.

Coding guidelines

Examples in public of rust projects applying rigorous change and configuration management, verification and documentation processes, including reasoning about how these processes achieve the objectives of safety standards, and evidence to support this.

Drive embedded platform development, support them so meeting regulatory requirements becomes easier, driver C++ interop

As safety is often done by imitation publicly accessible success stories and best practices help most.

Framework support, embedded Linux support

PowerPC, e200 cores are used heavily in automotive control. NXP5748, etc.

I believe it is time we sit at ESA's table and clarify the needs.
Waving Rust in space CAt. A SW in project shouldn't be something beinf frown at anymore...

What steps would your organization need to take to seriously consider or adopt Rust for safety-critical projects?

Number of responses: 46

Text answers:

Certified compiler, bare metal compilation.

We really need a certified Rust compiler for DO-178 DAL-D / DAL-C ASAP! This issue is currently the most critical one so Rust adoption can be performed in our company. If this does not happen in a foreseeable time I assume Rust is dead at our company.

Get the process in place and the get the certified compilers and tools to certify the Rust code

1. Demonstrateably

We _are_ using Rust for our product and intend to certify by end of 2025

1. Biggest hurdle: convincing management it is worth the cost of training. 98% of developers that I interact with (outside of our current c++ project) are c developers, and to train everyone in Rust is a really difficult sell (in terms of money and time).
2. Pressure from customers. We get paid by customers. They very often require either a c API, or they are just buying the entire solution (in which case they don't care about the language but more about the tools to configure the software). Since c already has established safety standards and guidelines that are meant to prevent most bugs, it is difficult to switch over unless a customer will pay us to do it.

We already use Rust for everything.

Establishment of safety compliance standards.

Being an ISO standard and being available multiple compilers, at least one certified

Allow developer teams to invest time into training and start with some smaller projects where it could fit. I think it just has to start somewhere.

Framework for 62304

1) Reduce scope for Rust to non robotics software infrastructure, or low robotics oriented usecases that don't require the numerical ecosystem

2) Come up with a reasonable way of interfacing between existing ROS system and Rust that is mature and stable (ideally Rust should be write once and forget for a while)

3) Justify to the team why Rust's features are better and worth investing in than just writing C++ more carefully. Also a maintenance plan to ensure that the module is maintainable with low language ability and doesn't break with Rust changes

Trainings
Adjusting processes

Training and establish processes related to Rust code quality

We have already done it, the project has officially committed to Rust, we're working with Ferrocene as tooling supplier and are transitioning into safety critical software development är the end of this year

I wish I knew that. :(

Customer demand

we are already in the process

The org would likely want some safety related coding guidelines and style guide to use

Improve bridge between Rust and ROS2

The Safety Critical Working Group is precisely the correct thing. :)

Regulatory announcement that forces our hand. Otherwise it is is both up like my department with tech stack standardization.

Trust and low project risk in switching

MC/DC Code Coverage tooling
Static Code Analysis guidelines and tooling
Code Metrics guidelines and tooling

Training, show evidence of lessening defects, integrate components with existing code base

We are actively taking steps now but this is new and uncharted territory ... any guidance from similar industries would be most welcome.

Prototype on a small scale project start to finish.

Would need to hire more experience Rust engineers and existing engineers need to learn Rust

We would need a "proved good" engineer for training of initial teams and setup of tooling according to

- widespread ability of engineers to hit feature goals with rust
- successful smaller projects with Rust (to convince remaining sceptics)
- Hiring: availability of experienced Rust devs for embedded is basically 0 (Northern Germany)
- knowledge of approval agencies accepting Rust usage

I think in general there needs to be better tooling to prove that rust code is safe. Currently we can easily prove memory safety but that is only part of the overall problem.

I'm sure that there are undefined behaviours within rust and ideally there would be tools that could detect code that contains undefined behaviour.

buy in from upper management, a proven in use set of tools, coding standards, ... that the approval body accepts

Management awareness about the benefits of using Rust.

Integration with ARM CC compiler

perform comparison against SPARK Ada and supporting tools

Significant restructure.

I don't think you guys understand functional safety, code is like 5-10% of the exercise, when done well, it is way more about hazard and risk, specifying requirements etc etc - rust or some formal methods language or whatever is no silver bullet. 95% of the problem is dredging out the actual problem and requirements from owners who think they know but are clueless. Tbh, I'd rather use a selected subset of misra c due to the extensive prior use in automotive, but I can't see hydrocarbons and mining clients going for any of it in the foreseeable future.

Greater emphasis from the government or community that keeps putting it in the c-suite people’s ears. Nothing will help like directives from the top

Presentation on the advantages of rust.

Funding and campaign for aerospace adoption. Get members from commercial companies on the Rust team, e.g. SpaceX, BlueOrigin.

Defined subset of RUST, verification tool vendor acceptance of RUST

There's just a lot of inertia to overcome

We have decided to go all in on rust for our main vehicle software which is specifically not safety critical (but if it fails, we lose a rocket/payload). We are looking at using baremetal rust for our flight termination unit in future flights where we get into regulations noted before.

Biggest need is finding industry partners to V&V our rust codebase when we go forward and start on our flight termination work.

We "simply" need to get one safety related project using Rust successfully to SOP, right now QNX8 support ended up in the critical path.

We already do

Integration with Simulink. Being able to take in that generated code as "unsafe" and work with it. Especially in control algorithm.

Tutorials on how to do it would be great if written by someone.

I am all in for join efforts, more ESA ITTs, common R&D projects and discussion !
Benoit LIETAER