C768 Task 2

Zachary J. Allen

Western Governors University

C768 Technical Communication

May 23, 2022

A. Executive Summary

A.0 *Rust in the Lab* Executive Summary

Writing Astoria in C/C++ is guaranteed to introduce crippling technical debt, which we won't have the resources to pay. This technical debt will take the form of memory and type safety issues across half a dozen platforms and perhaps dozens of interfaces. The cost to purge these issues is essentially infinite; permitting them will make Astoria flakey, inflexible, and obtuse. A strategic compromise will require innovative developers, driving up hiring costs. Rust is the best alternative. The Rust programming language guarantees memory and type safety across every platform Astoria visits. Rust is loved by those who use it, and is exciting to those who don't. Colombia Corporation can use Rust as part of our developer culture to attract the talent and inventiveness needed to thrive. Rust is the right choice for Astoria.

A.1 Tone

This executive summary is written from the perspective of the body of Colombia Corporation, utilizing '*we*' instead of '*I*' pronouns. C/++'s challenges are written with definite conjugations, choosing '*will*' instead of '*might*', and '*is*' instead of '*might be*' or '*may become*'. This was chosen to bear the authority and confidence supported by the evidence presented in the white paper, along with the use of a variety of sentence complexities and structures. The challenges are laid out in pair-structured sentences of increasing complexity, driving the cause and consequence of each weakness C/++ brings to the table. These pairs introduce the 'what' and 'so what' of each idea that will be discussed in later paragraphs. This is contrasted with the shortest two sentences of the summary ("Rust is the…"). Between these contrasts, sentences with more straightforward structure and more implicative claims lay the foundation of my argument for Rust in the Lab. This contrast is used to ensure the reader gets a sense of certainty and simplicity when considering Rust.

A.2 Jargon

The Computer Science jargon is limited to '*memory and type safety*', '*platforms*', and '*interfaces*'. Platforms and interfaces are a relatively foundational part of Astoria's value proposition, and their meaning in CS circles is very similar to broader use. For these reasons and for brevity, I let those terms go largely unexplained, relying on the business-oriented audience (likely the founders of Colombia Corporation) to have a sufficient understanding. I don't expect this business-oriented audience to know the minutiae of what '*memory and type safety*' means, which risks them disengaging with the text, but also invites them to continue reading to find answers. I took this risk because the details of '*memory and type safety*' are far less important than their economic implications, and omitting the details let me describe the implications in some detail while, again, maintaining brevity.

Because Rust is the subject of the white paper, I don’t consider it ‘jargon’, rather a technical term. To ensure the summary is understandable to those who don’t know what Rust is, I introduce it as a programming language, and omit all the details about Rust which aren’t necessary for the main points being made.

A.3 Timing

My summary is written to make as direct statements as possible, which is meant to reduce the risk of misinterpretation. Due to the technical subject of the decision at hand (‘Should Rust replace C/++?’), concerns about political correctness and politeness are minimal. That decision will likely take far more than a week and a few pages to make, which suggests the timing of the message depends on the macro timing of events at Colombia Corporation, rather than the day of week or time of day the message gets sent. With seed funding recently completed, it may be important to raise this issue quickly before the company begins growing its infrastructure, which lends itself to a meeting with my superior to discuss the decision-making process, and later meetings with the broader leadership team. Those meetings and this white paper contain some privileged information about Astoria, and could probably be presented to anyone who has an NDA with us, including consultants, investors, and other stakeholders.

B. Press Release

B.0 Rust in the Lab Press Release

Headline

Railings Installed at the Memory Hole: Rust Prevents Memory Safety Issues for Astoria

Location

Zach Allen, Software Development

Lead-In

Writing Astoria in Rust promises memory safety, squashing the majority of vulnerabilities in C/++ systems software.

Body

Writing Astoria in Rust promises memory safety, squashing the majority of vulnerabilities in C/++ systems software. These vulnerabilities seem impossible to shake: Microsoft found that, since at least 2006, "~70% of the vulnerabilities addressed through a security update each year continue to be memory safety issues." Other industry titans aren't spared either. When the patch for the 2015 Android Stagefright exploit was first released, replacing the original vulnerability with another one just like it, XDA-Developers.com wrote that preventing these issues in a lower-level language required developers to be "[...]more careful than the entirety of the Android security team, for sure."

What are these memory safety vulnerabilities? Essentially, they're mistakes in bookkeeping. Computers store every task they're working on in the system's RAM, which is like a massive whiteboard a mile long. One very important type of task is bookkeeping: a bookkeeping task assigns parts of RAM (sections of the whiteboard) to smaller tasks, so that none of the tasks write on or read from somewhere they shouldn't. However, if a bookkeeper is ever a little sloppy, an attacker can send them a tricky request, making the bookkeeper lose track of their work and let the attacker into parts of RAM they were never supposed to see. Now that the attacker can work outside their section of RAM, they might be able to mess with other tasks the computer is doing, like an online banking website, or an EMail manager. Suddenly, everyone you know gets a suspicious link to buy crypto.

Rust prevents these vulnerabilities by forcing tasks to follow a bookkeepers' code of conduct: each section of RAM must always have exactly one owner, can have either a writer or some borrowers, and can't be erased while borrowed. These seemingly common-sense rules for whiteboards are enough to make sure that tasks written in Rust can't have unnoticed memory vulnerabilities, making it a key choice in what language Astoria should invest in.

Contact Information

Zach Allen

Colombia Corporation

zacharyja@colombia.com

(509)555-2471

B.1-3 Tone, Jargon, and Timing

Evaluator, the press release and the FAQ have the same audience. I felt it appropriate to write them in the same tone, jargon, and with the same release timing. Please see the combined analysis below the FAQ.

C. Frequently Asked Questions

C.0 FAQ

Where can I learn more about the Rust language itself?

Rust has a website at https://rust-lang.org and an extensive FAQ at https://prev.rust-lang.org/en-US/faq.html. These include but aren't limited to the Rust language's design choices, how to get started, and detailed answers about achieving specific tasks in Rust.

Astoria's design is already under development. Isn't there a switching cost?

Yes, but only barely. Astoria's conceptual design is certainly under development, and some half-dozen C modules have rough prototypes, totaling at about 1,300 lines. C and Rust can both make use of each other through their Foreign Function Interface (FFI), with a small cost in performance and a small inconvenience for directly-impacted developers. This means, if necessary, these C prototypes can stay in the codebase and Rust can be built around them, with no benefit and no rewrite cost for their inclusion. Choosing Rust as the next iteration of these prototypes would introduce a small amount of rework, but no more than the inevitable rewrite prototypes go through anyway.

Won't Rust turn away experienced C/C++ developers?

It's nearly guaranteed that some developers will miss our job listing because it's for Rust, not C or C++. It's likely that some developers will find and ignore our job listing because it's for Rust. It's reasonable that these missed developers have experience and skill. These consequences will accompany the huge developer interest in writing and learning Rust, which selects for developers who want to try new things and lay stable foundations.

C/D Analysis

C/D.1 Tone

The audience for my informative publications is the whole of Colombia Corporation, including those without any technical expertise. This led me to a more relaxed and educational tone, relying on an extended metaphor and personification to help ease the acceptance of the underlying technical ideas. This tone is not perfectly consistent, where the FAQ section tailors the tone of the second question to someone with some development experience. The goal of these publications is to subtly introduce the central ideas of the white paper, while setting up the talking points I expect will come to discussion throughout the company.

C/D.2 Jargon

With the exception of the second FAQ, I went out of my way to avoid or explain any jargon used; so, I'll discuss the second FAQ here. I rely on the reader having experience writing and planning to write code, and some understanding of the common practices of prototyping a piece of software. Specifically, I use ideas like 'C module', 'lines of code', 'Foreign Function Interface', 'codebase', 'rework', and 'rewrite', all of which should be entirely familiar to anyone who has been a part of any software development project. This jargon is left unexplained because the real purpose of answering the question is to show project sponsors that my suggestion to switch to Rust has been well thought out and balanced against the switching costs, and to offer them evidence of that conclusion.

C/D.3 Message Timing

Releasing these publications before having any meetings with my supervisors about the white paper would undermine the decisions my supervisors have a right to make at those meetings. So, I would wait until the Monday or Wednesday morning after those meetings to publish these. The goal would be to help the general employees of Colombia Corporation raise any concerns or questions they have with their supervisors, so those questions can be filtered up and eventually addressed as the decision-making process goes on.

References

Miller, M., & Microsoft. (2019, February 6-7). *Trends, Challenges, and Strategic Shifts in the Software Vulnerability Mitigation Landscape*. YouTube. Retrieved May 10, 2022, from https://www.youtube.com/watch?v=PjbGojjnBZQ

XDA-Developers.com & Pulser_G2. (2015, August 19). *A Demonstration of Stagefright-like Mistakes*. XDA-Developers. Retrieved May 15, 2022, from https://web.archive.org/web/20150819162559/https://www.xda-developers.com/a-demonstration-of-stagefright-like-mistakes/