Custodio, Rommel Garcia

Yokohama, Japan
sessyargc.jp@gmail.com • LinkedIn

AT A GLANCE

- Embedded system software developer
- Possesses hands-on knowledge on the use of hardware tools (signal/protocol analyzers, logic probes, oscilloscope and JTAG debuggers), can solder and understand schematics
- **Also** possesses hands-on knowledge on the use of OSS software tools for static code analysis, code formatting, sanitizing and debugging
- **But**, would rather create simple efficient code than debug
- **Always** utilizes the test driven approach to development
- Innate passion to know how things work
- Fast learner and critical thinker
- Possesses a Bachelor's degree in Mathematics (majoring in Computer Science)
- Capable of working independently as well as part of a team
- Does not need an IDE to get work done, give me vi/vim/neovim and ksh
- **work is play** attitude ever since graduating from university
- **In my spare time I**: code, study/learn, tinker, read, explore, break things (*sometimes*)

GOALS

- Apply my **experience** and **expertise** to new technology domains
- Use the latest tools and techniques to implement efficient, safe and secure systems that will benefit society

SKILLS, EXPERIENCE

- Specialties: Embedded software and real-time systems; Operating systems; Test automation; Board bring-up; Bare-metal embedded development; Open Source Software (OSS)
- Operating Systems: **Linux** (Ubuntu, Arch Linux, Alpine Linux, chroot); OpenBSD; Windows (WSL/WSL2);
- Single Board Computers/MCUs: Raspberry Pi; Raspberry Pi Pico (RP2040); pyboard (STM32); Milk-V (RISC-V); 8051; Z80
- Programming Languages: **C/Modern C** (25+ years, **C99**, **C11**, **C18**, **C23**); **C++/Modern C++** (5+ years, **C++11**, **C++14**, **C++17**, **C++20**, **C++23**); **Python** (5+ years); **Rust** (1 yr, self-study); **Haskell** (1 yr, self-study); **Swift** (self-study); **Hylo** (self-study); Assembly (ARM, RISC-V)
- Version Control System: **git** (10+ years); **CVS**; **GitLab**; **GitHub**; Perforce
- Virtualization/Emulation: **Docker**; **QEMU**
- General tools: vi/vim/neovim; ack/ag/grep/ripgrep; sed; awk; Korn Shell; make; cmake; gcc/g++/gdb; gcov/lcov; clang/clang++/lldb; clang-format; clang-tidy; cppcheck; libFuzzer; valgrind/memcheck; ghc/ghci; cargo/rustc; Codeium; GitHub Actions; GitHub CodeSpaces; VS Code (Live Share); Google Test; Google Benchmark

WORK EXPERIENCE

Kyocera Document Solutions Minatomirai Research Center, Yokohama, Japan

I transferred to Japan from the US, the company renamed and moved to Yokohama in 2018

- Software Engineer, Research & Development 2008 – present
  - Studying modern static code analysis using CodeQL and LLVM.
  - Successfully converted PoC scripts (small, <1 KLOC, duck-typed scripting language) to C++ (modern C++ is only C++11 in this case because of integration requirement ... two dreaded words, legacy code).
  - Successfully integrated Python code to C# using (**PythonNET**).
  - Successfully implemented an automated data acquisition/scraping system in Python, and later converted to Rust as a proof-of-concept and programming practice.
  - Successfully implemented an on-premise automated integration build and deployment system using **Jenkins**, **Docker** and robotframework.
  - Successfully implemented a chat-based control system for remote automated device power control using **Python**.
  - Successfully converted an internal image comparison system that initially used pixel-by-pixel comparison to use a Traditional Supervised Machine Learning model using **scikit-learn**, **metric-learn**, and **SKLL**.
  - Successfully constructed Compute Engine (GCE) virtual machines on Google's cloud platform for use in Machine Learning investigations.
  - Successfully implemented an image acquisition, analysis and verification system using **Python**.
  - Self-study of **AutowareAuto** course (ROS 2, Foxy) because of my interest is **DDS** (Data Distribution Service).
  - Successfully converted an internal system to use ROS DDS as the data transport subsystem.
  - Ported, implemented and tested Linux-based software written in **C**/**C++** for PowerPC and ARM architectures.
  - Performed successful board bring-up of new platforms using **u-boot**.

- **Linux** OS kernel maintenance, back-porting latest mainline patches to internal development branch.
- Investigated and fixed reported bugs.

Kyocera Technology Development, California, USA

- Software Engineer, Embedded Systems Engineer 2006 – 2008
  - Ported, implemented and tested Linux-based embedded software written in **C/C++** for printer controllers.
  - Optimized proprietary image pipeline using multi-core processing.

Canon Information Technologies Philippines, Manila, Philippines

- Software Engineer, Technical Lead, Quality Assurance Specialist 1997 – 2006
  - Designed, implemented and tested embedded software written in **C** for printer controllers.
  - Supported the design, testing, bench-marking and conformity certifications of the project.
  - Attended Bluetooth UnplugFest (an international interoperability testing event) organized by the Bluetooth SIG.
  - Monitored discussions of the Printer Working Group (PWG).

EDUCATION

University of Santo Tomas, Manila, Philippines

- BS in Mathematics Major in Computer Science 1994 – 1997
  - Thesis: LUCas Encryption
  - Focus: Implementation of LUC encryption based on a Dr. Dobb's Journal article *LUC Public-key Encryption: A Secure Alternative to RSA* **1993**, public key cryptography.

CONTINUOUS LEARNING

ACCUConf	**CppCon**	**CppNow**	**cpponsea**	**LLVM**
MeetingCPP	**NDC Conferences**	**C++ reference**	**C reference**	**Udemy**

LinkedIn Learning

RIKEN Center for Advanced Intelligence Project (AIP) English Presentations

- RIKEN AIP Youtube Channel

MIT OpenCourseWare

- 6.5940 TinyML and Efficient Deep Learning Computing
- 6.034 Artificial Intelligence

Cornell University

- CS4780 Machine Learning for Intelligent Systems

LANGUAGES

Filipino/Tagalog/English: **Native level**
Japanese: **Greeting level**

INTERESTS

old computing/vintage devices, Open Source Software, Operating Systems, Programming Languages, Secure Programming, Functional Programming