James Grant

E-mail: james.grant@outlook.co.nz
Mobile: (021) 106-5479
Work Rights: NZ Citizen
Current Location: Christchurch

Summary

- Experienced (16 years) software developer.
- Primarily developed in C and C++ (up-to 2017 standard) languages. In C++ I make use of object orientation, exceptions, STL, RAII - i.e. not using it simply as a 'better C'.
- Windows desktop application development – became sole developer/maintainer of fire panel GUI configuration software written in Embarcadero C++Builder. Have also made use of Visual Studio.
- Embedded Linux – application level (C++), as well as BSP development – bug fix and integration work for kernel and U-Boot, image and toolchain generation with buildroot.
- Bare metal embedded – developed for small 8-bit microcontrollers through to larger 16/32-bit ones where more sophisticated multitasking is employed. Have the ability to write low-level hardware interfacing code, read schematics, and use electronic test equipment.
- Proficient in Python for automation tasks or test applications.
- Use of build, debugging and diagnostic tools such as CMake, Cppcheck, gdb, sanitizers, and Wireshark.
- Carried out revision control using Git hosted by GitHub.
- Use of JSON and XML libraries for storage or interchange – both DOM and SAX style APIs.
- Performed security and open source assessments, made use of standard Crypto APIs.
- Mostly worked in a smaller team environment (sole developer on a project in a multi-disciplinary team, or in a team with a few other developers) – tended to have ownership of functional area(s).
- Worked at all stages of development cycle – design/specification, estimation, implementation, test planning/execution, writing of end-user documentation, reporting project status to management, and the handling of in-field issues elevated by support staff.
- Support and mentoring of staff with new development tools.
- Use of Jira for issue tracking and project management.

Employment History

Software Engineer — *Nov. '07 – present*

Johnson Controls – Fire Detection Products (Christchurch)

Johnson Controls develops fire detection equipment – fire alarm panels, detectors, evacuation systems, brigade signalling equipment, and other associated ancillaries.

Specific work I have performed in this role includes:

New Evacuation Panel: Implemented custom USB transfer protocol for firmware and configuration transfer to panel – utilised user-space FunctionFS gadget on Linux device side, and driverless WinUSB API on PC side. Re-implemented and enhanced scripting language from previous generation product using a recursive descent parser and bytecode interpreter. Configuration tool development on PC side, loading of JSON configuration on device side. Shared implementation work of main panel user space application. Adapted the SOM-specific Linux kernel and U-Boot boot loader – bug fixes, integration of patches, device initialisation changes, and device tree config. Setup image and toolchain builder using buildroot. Support and mentoring of other staff with the development environment.

PC Configuration Software: Became sole developer/maintainer of PC based configuration tool for fire and evacuation panels written in C++Builder. Implemented UI & logic changes as new functionality required. Where responsible for accompanying firmware development, proposed UI & logic changes, and consulted with other staff. Worked with in-field and support staff to resolve problems. Transitioned installer to Inno Setup for ease of scripting.

Support for New Detectors & Ancillaries: Carried out specification and implementation of fire alarm panel firmware for the company's *MX* digital addressable detectors (smoke, heat, CO, flame) and ancillaries (manual call points, conventional I/O, 4-20 mA, beacons, sounders). Implementation areas included communications signal decoding, protocol message handling, sensor data processing, and other logic. Wrote configuration tool support for the new devices. Collaborated with overseas R&D centres responsible for their development. Prepared and supported submissions for type approval testing.

Addressable Phone: Initial demonstration of capability using Power Line Communications development boards. Achieved group call function with mixing of AMR-NB codec audio frames broadcast using G3-PLC on a shared bus. Python script used to interface with PC audio over USB serial connection.

VoIP Streaming Solution: An obsolete Cisco ATA (analogue telephony adapter) used in an emergency evacuation system supported one-way streaming from a single audio source to ten callers for PA, warden phones, background music, etc. The replacement adapter supports only four, which is inadequate. To provide a solution, a VoIP streaming proxy was developed and integrated into an existing IP bridge product. An independent light-weight SIP stack was developed that handled receiving calls from other nodes and making calls out to a single local ATA. The streamed audio frames received from the ATA using RTP protocol are simply forwarded on as-is to the proxy callers.

Enhancement of Network Router: Implemented enhancements to firmware of a proprietary network protocol router to improve ease-of-use. The product is primarily used for RS-485 ring (dual path) networking of fire alarm panels, although other topologies (e.g. multi-drop, point-to-point) and media (RS-232, TTL) can be utilised. With the enhancements, a common scenario now requires zero configuration, for other scenarios CLI improvements and automatic address learning largely eliminates tedious configuration previously required at each node. PC based message generator software was developed in C# for load testing.

Increase of Fire Alarm Panel Capacity: Worked with others to optimize fire alarm panel firmware and that of the connected detection loop interface cards to support more detectors simultaneously. For performance analysis and testing, I designed and implemented software to simulate the communications ICs of a large number of detectors. Later performed work to add a fan control board to the same bus as the detection loop interface cards.

Apartment Module: Wrote software design documents and developed firmware from scratch. The module interfaces with the fire alarm panel, detectors, and sounders so that smoke generated activations are local to the apartment and hushable. All other activations (e.g. heat) lead to evacuation and calling of the brigade.

Utility Software Enhancements: Made improvements and fixes to PC based utility programs used for quoting production jobs, fire alarm panel remote access, and system power calculation. These are written in C++Builder or Delphi.

Education

2003 – 2006 **Bachelor of Engineering (Electrical & Electronic) with Honours**
University of Canterbury