Raymond Gauthier

linkedin.com/in/jraygauthier

Education

2004-2008 **Bachelor's Degree, Computer Engineering**; Sherbrooke University (Sherbrooke)

Robotic & AI concentration

2002-2004 **College Diploma, Pure And Applied Science**; Cégep Lévis-Lauzon (Lévis)

Work Experience

Amotus Solutions / Dimonoff (Québec)

Senior Software Developer, Sept 2017 - Present

Client 3 (Sep 2019 - Today)

An embedded power metering device running multiple Dbus interconnected Python3 services hosted on a custom Yocto Linux distribution that pushes its acquired data cloud side through a google cloud platform MQTT channel taken over by a JS cloud function that stores data to both Firestore and BigQuery. A cloud side PHP dashboard manages / exposes / leverage device metering data. Devices are individually configurable through a local web UI accessible via: USB Gadget, Ethernet and recently even a ZeroTier VPN.

Responsibilities:

- Technical SW lead, release manager, sprint planning and lead.
- Taking technical requirements from the client, proposing plans to carry out these requirement and finer grained splits and evaluations.
- Take over device's Yocto configuration evolution.

Major achievements:

- A nice CLI application based written in Rust and cross compiled from Linux to Windows to simplify the deployment of custom Yocto distributions to target embedded devices. Tool shipped via a Linux produced `*.msi` installer, thanks to nice open source `msitools` project.
- Design and code a simple web UI app (Aiohttps, Jinja2) with login to configure devices.
- Design and code a socket and generator schedule service with a frontend component added to the above web UI.
- Reproduce embedded environment as faithfully as possible on developer Linux PC through a reproducible Nix environment not only including Python dependencies but also non-Python system dependencies.
- Various minor features additions and bug fixes.

- Incremental redesign or the numerous services (taken over from the initial prototype).
- Incremental code base modularization, moving service common code to a common well-designed modular library package.
- Incremental code base *type checking* using mypy.
- Increase embedded backend services coverage to ~90% allowing both fake (on pc) and real HW test (on embedded) testing even dbus service inter-communication.
- Migrate project packaging from Setuptools to Poetry.

Technological Environment: Python 3 (Pytest, Mypy, Asyncio, Aiohttp, Jinja2), Rust, Yocto, Dbus, Systemd, USB gadgets, Html, JS, Css, Docker, Docker-compose, Google Cloud Services (Cloud functions, Firestore, Big Query), PHP, PostgreSQL.

Client 2 (Sep 2020 - Dec 2020)

A client internal inventory management and tracking system implemented as a C# Blazor app working with MS SQL Db data through a well-designed C# MVC RestAPI. According to client specification, Amotus exclusively designed and developed an embedded C# Xamarin bar code scanner application connected to the same RestAPI facilitating inventory product management / manipulations.

Responsibilities:

- Work with client and client in-house developer / architect to improve upon a C# MVC / Blazor app prototype they developed.
- Work on an agreed upon set of tasks, refining requirements and weekly demoing results to client.

Major achievements:

- Document, design and implement first best practice example of form error validations leveraging FluentValidation, MiddleWare and Blazor so that backend errors properly bubble up to user (on a per field basis when the case) and that real-time frontend side validations in Blazor C# are made possible.
- Design and implement proper internationalization of the overall application so that language selection made through the Blazer App is made available to the separate MVC App implementing the RestAPI.
- Design and implement proper user Authentication and Authorization system both preventing unauthenticated use of the system and allowing access to current logged-in user in all layers (Blazor browser/backend side and MVC app). RestAPI protected using JWT Token. All this done leveraging at all layers following new .NET core best security practices.
- Understand and document tooling to perform by the books migrations of existing EntityFramework MS SQL database supporting colleague with first required migration.
- Adapt existing windows only application so that it can easily be run and developed on a Linux PC nix reproducible environment, leveraging new .NET Core Linux support to its full extent. Includes proper reproducible VSCode setup with `ms-dotnettools.csharp` based on `OmniSharp`.

Technological Environment: C#, .NET Core, Entity Framework, MVC RestAPI, Blazor, FluentValidation, MS SQL, Docker, Docker-compose, Nix, C# Xamarin.

Client 1 (Sep 2017 - Oct 2020)

A medical device running a client (JS) / server (async Python) application hosted on a custom Linux / Nixos distribution. Overall goal was to configure a MCU trigging a camera, a spectrometer, a laser and some LEDs collecting the camera / spectrometer frames trough usb making sure not to lose sync. Collected frames would then be bundled as whole cycles and send in real-time to the JS frontend through websockets while being persisted for later sync with the cloud storage service.

Responsibilities:

- Taking technical requirements from the client, proposing plans to carry out these requirement and finer grained splits and evaluations.
- Architecture and code a sub-system orchestrating the overall acquisition process of the system reconciling various sensors including a framework allowing client's own developers to safely develop sequenced or parallel asynchronous real-time high-level algorithms operating on the acquired data with two possible kind of effect: sensor data alteration and actuator control.
- Custom Linux distribution and tools to factory install the devices with the custom distribution. These tools include a bi-directional update system allowing secure deployment of secrets and much more.
- Multiple cli helper utilities program for semi-automated system administration, factory install and support. Tools were developed both in bash and python and exposed through `PATH` via a nix package / developer environment.
- Reproducible development environment using the nix package manager.
- Thorough integration tests for the overall acquisition process using pytest.
- Very precise unit tests for both acquisition devices leading to multiple fixes by third parties (TIS and Hamamatsu).
- Multiple improvements to the backend application, particularly in terms of real time and offline configuration for the acquisition process.
- Migrate existing application from python2 to python3.
- Add optional static types to the code base using mypy to improve scale and maintainability of the project.
- Add missing packages to the open source nixpkgs package set and improve existing when required.
- Documenting the acquisition process, the custom distribution and the development tools required to work with the project.
- Document restAPI requirements for the IoT cloud storage service using Haskell servant library and swagger so that chosen third party can provide required end points to store the acquired data.

Technological Environment: Python3 / Python2 (Mypy, Pytest, Asyncio, etc), SQLite, Haskell (servant DSL), Swagger, Html, CSS, JS, AngularJS, Linux Nixos, Bash , Gnu Tools, Nix, Nixpkgs, Git, Markdown (pandoc), Sphinx.

Innovmetric (Québec)

C++ Software Developer, Aug 2016 - Sept 2017

- Improving a complex existing metrology application used by most if not all car / plane companies according to spec (including challenging the spec) in 7-9 scrum teams. Very well implemented agile methodology (including the developed feature presentation) and overarching strategy to maintain a huge code base by huge team of developers. Thorough integration testing strategy through software specific macro system.

Technological Environment: C++14, Windows, MFC, Mercurial, Visual Studio 2015, huge set of private libraries.

Ungava Technologies (Québec)

C++/Qt Embedded Software Developer, Jun 2013 - July 2016

- Developed an optional dynamically loaded plugin system to allow our clients to easily extend our embedded application with their own *C++/Qt* views and algorithms. Designed to provide strong backward source/binary compatibility guarantees and maximize possible customizations.
- Direct contact with the client, multiple design presentations, announces of new feature, in depth technical explanations, details evaluations and independent project management.
- Modularized / refactored an initially non-modular *C++/Qt* executable to accelerate developments.
- Design of multiple public and private libraries (e.g.: missing standard library utilities, cartesian geometry, sensing types, configuration, hardware capabilities description and many others) all with their own unit tests and documentation.
- Add multiple new feature to the system through agile methodology through heavy use of unit tests and when necessary using gui experiments.
- Fixed multiple bugs.
- Thorough testing of the embedded device using full *NDT* setups (probe, part, scanner, etc). Description of the observations, assumptions and hypotheses.
- Augmented Qmake build system so that it is easier to work with multiple library dependencies (>30). Deferred to some python scripts for dependency order resolution.
- Augmented our development machines with *Nix/Nixpkgs* so that our team's development configuration can be reproduced and customized robustly.
- Developed a customized documentation system on top of *Doxygen* that can work in a modular way and allows references between documentation components.
- Developed a method to facilitate development across multiple repositories of any types (now mostly using *Git*).

Technological Environment: C/C++, Qt, QMake, Embedded Linux (Yocto), NDT systems, Doxygen, Markdown, PlantUML, Graphviz, Haskell Diagrams, Latex Math, Pandoc/Gitit/Hakyll, Mediawiki, QtCreator, Ubuntu/Nix/Nixos, Bash / Gnu Tools, Haskell, Python, Html/CSS/Javascript, Git, Svn, Bitbucket, Jenkins, Apache

Bentley Systems, Inc. (Beauport)

C++ Software Developer, Jun 2009 - Jun 2013

- Increased georeferencing and reprojection facilities by adding support for WKT standard fitted CS and local CS
- Created an import framework supporting multi-layered, georeferenced, multi-dimensional and polymorphic data
- Storage and import of multi-resolution and single-resolution DTM data
- Increased STL functionality
- Increased smart pointers functionality and security

- Created ATP tests for ImageLib API in order to preserve its functionality when core was replaced by ImagePP
- Created an ATP framework designed to facilitate comparison of actual system results with a specified baseline
- Image processing (tiling, stripping, filtering, etc)
- File formats support

Technological Environment: C/C++, ImagePP framework, ATP framework, Visual Studio .NET 2005/2008, Microstation, Descartes, WinCVS, Bentley Build, Mercurial

Bluberi (Drummondville)

C Software Developer, Mar 2009 - Jun 2009

- C Development of 2D casino games optimized for company specific boards.
- Undertook the whole formation/training in order to be able to perform the task.

Technological Environment: C, Visual Studio .NET 2005, Custom Casino Boards

Bentley Systems, Inc. (Beauport)

C++ Software Developer (Intern T4), May 2008 - Aug 2008

- Develop/integrate new file formats in ImagePP framework
- Fix/adapt ImagePP integration as new core for ImageLib API so that the library behave in at least the same way or better than former core
- Fix multiple artefacts/bugs (TRs) in Raster Manager and Descartes products
- Attend courses on 3d design with Microstation(tools usage guidelines, rendering, new functionalities to be delivered with next release, etc.)

Technological Environment: C/C++, Visual Studio .NET 2005, Microstation, WinCVS,

CAE Inc. (Montréal)

C++ Software Developer (Intern T3), Sep 2007 - Dec 2007

- Develop/integrate/test new visual functionalities (new specs for luminous points display, geodetic positions selection, etc.)
- Fix multiples artefacts/bugs of greater complexity

C++ Software Developer (Intern T2), Sep 2006 - Dec 2006

- Analyze / apply solutions for reducing application memory footprint and raise actual limits
- Develop/design an API enabling memory footprint reports for individual modules of the application layer
- Fix multiple artefacts/bugs on the graphical application

Technological Environment: C/C++, Java, Visual Studio .NET 2003, Borland JBuilder, Flight Simulators, Multithread programming, Shared Memory, Unicode vs ANSI, StarTeam, Visual Basic Application, Smart Heap

Promutuel (Québec)

Java Software Developer (Intern T1), Jan 2006 - Apr 2006

- Develop servlet-based applications following specifications from a functional document
- Develop embedded web scripts
- Perform unit-tests

Assistant to the re-engineering of business processes (Intern T0), May 2005 - Aug 2005

- Help/guide remote distance employee with their PC related problems
- Design/develop interactive forms
- Compile orally specified or written procedures using UML notation

Technological Environment: Java 1.4.2, WSAD (Eclipse 2.0), Struts-Tiles Framework, JUNIT, JSP, JavaScript, Adobe Designer

Technical Experience

Work on my personal private cloud

Hosted on **Amazon EC2** through multiple **Nixos** Linux machines, 2015-2016

- Declarative deployment using *Nixops*
- Declarative machines configuration using *Nixos/Nix*
- Static code serving, proxying using *Nginx*
- Static website compilation from *Markdown* using *Hakyll*
- Dynamic textual data provisioning using *Git* repositories and *Systemd units* dependencies to ensure data availability prior to dependent units / services.
- Custom private *Markdown* wiki using customized *Gitit* through *Haskell* plugins for figure rendering (e.g.: PlantUml, Haskell Diagrams, etc).
- Added a modern *javascript* frontend for wiki page edition to my gitit instances that provides side-by-side editor / preview pane with fast automatic html preview rendering.
- Currently working with *Haskell Servant* library to develop a nice statically typed web api. Also evaluated *Haskell Yesod*.
- Currently use a secure communication channel through *HTTPS* using a *self-signed certificate*. However, I'm about to integrate with the *Lets Encrypt* system in order to get *full-fledged and free certificates*.

Open Source

Nixpkgs: Multiple contributions to the default package/configurations set of Nix/Nixos, 2015-2021

All of my personal and work computers are now using *Nixos*. Over the time, added multiple packages to *Nixpkgs* and fixed multiple issues. Doing this made me pretty fluent with the open source development process.

Programming Languages

Python:

Worked with libraries: asyncio, tornado and many others.

C++:

Worked with libraries: Qt, boost, wxWidget, OpenGL, MFC and many others.

Nix:

Build many tools using nix over the years in order to make builds reproducible anywhere.

Haskell:

Worked with libraries: servant, pandoc, diagrams, wx-haskell, reactive-banana, turtle, yesod, happstack, aeson, pipes and many others.

Good knowledge of: Javascript, Bash, Java, Matlab

Basic knowledge of: Python, Visual Basic, Cuda

Development environment

OSes:

Linux, Windows

Build systems:

Qmake, gnu make, CMake, cabal

VCS:

Git, Mercurial, Svn, Cvs

IDE:

Vscode, QtCreator, Microsoft Visual Studio, EMacs, xCode, KDevelop, Eclipse, JBuilder

Continuing education

Functional programming related:

- Reading of many Functional Programming / Haskell papers, 2014-2016
- Haskell custom file format conversion tool using parsec, 2014
- Reading of “Haskell school of music” book and experimented with examples, 2014-2015
- Reading of “Real World Haskell” book and experimentation with its examples, 2014

Object oriented programming related:

- Read multiple C++ books about good practices and design, 2006-2014
- OpenGL learning project reading the “Red Book” (C/C++), 2007-2008
- Custom parallel port activity logger (C++, MFC), 2006-2007
- Create some new Half-Life2 entities with custom behavior (C++), 2006
- Self-taught object oriented C++ learning, 2002
- Self-taught Visual Basic, 2000

Miscellaneous:

- Coursera - Critical Thinking in Global Challenges course, 2013

University projects

Neurospike Project, 2008

C++, optimization of the C++ reference implementation of an image recognition neural network using a NVidia tesla server (Cuda) and development of a real-time debugger / analyzer of its internals (wxWidget and OpenGL).

Design of a distributed system, 2007

Java, RMI, Architecture MVC

Design of an embedded system with networking, 2007

C, uC-OSII (RTOS), TCP/IP stack, protocole HDLC RF link, interrupts, UARTS, microcontroller interfacing, timers, ARM architecture, cygwin

Design of a customized drink distribution machine, 2006

Java, Cervelets, Tiniboard (embedded system), network, electronic interfaces, fully functional prototype

Conception d'un brouilleur de signal, 2005

C++, XILINX, data transfer protocol, logical gates et Qt GUI

Conception d'un robot interagissant avec son environnement, 2004

C, Handiboard, electronic interfaces (lecteur DC et autres)

Miscellaneous

- Certifications and Professional Memberships *Ordre des ingénieurs du Québec* Received on Sep 23, 2009. Ended membership since 2014.
- Spoken/written languages:

  French (5/5), English (5/5)