Daniel W. McRobb

Clarkston, MI 48348
mobile (248) 622-8246
home (248) 605-8500

dwm@mcplex.net
dwmcrobb@me.com

Professional Experience

30+ years professional software development: 15 years in network management and measurement software, 14 years in embedded software, 2 years in vision system software. C++ programmer since 1996, C programmer since the 1980's. UNIX application programmer since 1990.

ONE — 2022 - 2023

Senior software engineer

Battery management system: Developed safety software for SPC58 microcontroller. Developed FCCU (Fault Collection and Control Unit) self-tests. Configured STCU (Self Test Control Unit). Aided in safety analysis for ASIL rating.

UDS (Unified Diagnostic Services): Bootloader work for TI C2000 used in DC-DC converter. Finished basic ISO UDS stack for reflashing over CAN. Implemented UDS stack on Linux for Raspberry Pi that allowed speedy reflashing of 16 converters over CAN.

Battery topology discovery/display: Worked with a small team to develop a means of discovering and displaying battery system topology (series/parallel), a textual representation that could be parsed by humans and machines, and a simple Qt graphical program to display the topology graphically.

Mentoring: Mentored and trained young developers on C programming language.

Casco — 2021 - 2022

Senior software engineer

Engineering evaluation for high-power DC to AC inverters: Developed scheme to run multiple inverters in parallel to achieve higher output current. Developed an improved spread-spectrum PWM implementation to reduce radiated emissions. Evaluated, documented and presented NFPA requirements for inverters in different contexts (jobsite versus residential power, etc.)

.

ASPICE audit support: Provided documentation for ASPICE audit.

2kW DC to AC inverter software: Assumed release engineer role for 2kW inverter for Ford F150. Bug fixes, minor feature additions, test plan development and execution for the main microcontroller (S32K). Support for Ford PARSED data and testing.

LEONI — 2018 - 2020

Software engineer, team lead

Vision inspection systems: Vision inspection system development in C++ with various technologies: Qt for user interface, Matrox Imagine Library (MIL) for camera communication and image processing, Cognex ViDi for deep learning, Google protobufs, JSON, etc. All on Windows with Microsoft toolchain and LLVM toolchain (code is portable to Linux). Inspection of various aspects of wheel and tire assemblies, injection molds, sheared/punched steel parts, etc.

Team lead: Planning, scheduling, mentoring, design, functional requirements, code reviews, test plans. Scrum master, phabricator administrator, release engineer. Instituted unit testing using Boost Test. Created frontend processes (requirements, design documentation), backend processes (testing, release process, issue tracking). Instituted scrum.

Unnamed startup (in stealth mode) 2017
Software consultant

Network security: Application security using Crypto++ and extensions to libDwmAuth. iOS, FreeBSD and Linux integration.

asynchronous DNS: Asynchronous DNS resolver (from scratch) and TCP connect wrappers in C++ for 'Happy Eyeballs' (RFC 8305) implementation. iOS, FreeBSD and Linux integration.

Kostal 2013 - 2017
Senior Project Engineer

Door control unit: Embedded software in C for a DCU with window and other door controls. Sensorless, pinch protection, etc. Renesas RH850 and RL78 microcontrollers.

Electronic shifter: Developed software in C for an electronic rotary shifter for a new Ford vehicle with ISO 26262 safety requirements. Supported SPICE/CMMI audit. Co-developed software validation and integration test plans. Project utilized Renesas 32-bit (V850) and 8-bit (78K) microcontrollers.

unit test skeleton generator: Used LLVM/clang frontend APIs to create a tool to generate unit test skeletons from existing code. This saved a lot of time in creating unit tests for code we ported and modified for various projects on various microcontrollers. Modern C++ used (C++11/14).

CANdb parser: Created a tool to parse Vector CANdb files to insulate our projects from database changes made by our customers. Tool generates a fully documented header file for inclusion in our source. Modern C++ used (C++11/14).

CAN log parser: Created a tool to parse CANoe log files. Created for a project using CAN-TP, with a complex application layer protocol above the TP layer (infotainment in a large luxury vehicle with multiple graphical displays and sources).

S-Record parser and CRC embedder: Created a general purpose C++ library and command-line tools to process S-record files for microcontrollers. Integrated into our build and debugging environment.

GM Research via MicroMax and Danlaw 2013
Senior Software Engineer

Autonomous vehicles: Developed software in C++ for use in autonomous vehicles, aggregating and distributing data from various sensor sources.

Lear Corporation via MicroMax and Danlaw 2008 - 2013
Senior Software Engineer

EVSE: Sole developer of software for all 2012 and 2013 Lear EVSE: custom bootloader, applications (12 variants) and diagnostics. Developed custom protocol and PC-based GUI application to communicate with bootloader for reflashing, fault log retrieval and software identification. Developed diagnostic display on my own time, which greatly reduced the man hours required for design and production validation. Prepared all documentation required for UL1998 certification (risk analysis, architecture document, etc.). Developed the software validation test plan. Provided input to hardware design. Developed validation test box for new features and end-of-line testing.

2011 Chevy Volt on-board battery charger: Developed the bootloader and application for one of the three microcontrollers in the 2011 Chevy Volt's on-board high voltage battery charger. Designed the messaging protocol (running over SPI) used between all of the microcontrollers. Implemented the messaging protocol on two of the three microcontrollers.

2013 Smart EV battery charger: Designed and implemented the bootloader for two of the three microcontrollers (Freescale S12P family). This was a tight deadline project. I delivered the code and documentation in 9 working days.

Arbor Networks 2004 - 2008
Senior Software Engineer

Addressed scaling issues in Arbor's SP (Service Provider) software suite (a NetFlow-based system). Introduced C++ and CORBA to the system (migrating away from CGI for middleware). Helped port the system from OpenBSD 2.8/3.3 to OpenBSD 3.6 with SMP. Updated the toolchain to gcc 3.4.3 (from gcc 2.95). Removed bottlenecks in distributed messaging and relational database transactions. Integrated sFlow as a data source.

Trendium 2004
System Programmer

Helped port Trendium ServicePATH product to Linux from Solaris. Project completed at the end of June 2004.

Ixia 2002 - 2004
Chief Architect, NetOps

Chief architect of NetOps suite. Continued work on NetFlow and BGP-4; Caimis was acquired by Ixia in late 2001, renaming the product suite `IxTraffic`. Helped develop reporting GUI for `IxTraffic` using java and CORBA. Implemented compressed storage of raw NetFlow data for detailed accounting. Improved performance of BGP-4 implementation in `IxTraffic`. Added new types of aggregate data to `IxTraffic`. Started design and implementation of configuration system for all Ixia products, using XML Schema and DOM (using `Xerces` from the apache group).

Caimis 2001

Co-founder, Chief Architect

Developed systems for collecting and correlating NetFlow data from Cisco and Juniper routers with BGP-4 data for the purpose of traffic engineering in large networks. Developed modern BGP-4 implementation (C++, multithreaded) for passive monitoring of BGP-4 from any number of peers. Used CORBA as the middleware, permitting remote retrieval of routing tables, route lookups, etc. Served as chief architect for entire Caimis software suite. Provided customized solutions and support for customer-specific analysis of BGP-4 within their networks (MED cycling, etc.). Developed build and upgrade deployment strategy for distributed data collectors, and distributed monitoring of the collectors (decentralized).

CAIDA 1998 - 2001

Researcher, Software Engineer

Developed `cflowd`, a system for collecting and analyzing data from Cisco NetFlow output. Developed `skitter`, a large-scale Internet path measurement tool using ICMP. Developed `arts++`, a library for storing and retrieving large quantities of data from `cflowd` and `skitter`. Helped Cisco develop NetFlow version 8, and helped Juniper test their NetFlow implementation. System builder and administrator for CAIDA measurement hosts.

ANS (Advanced Network and Services), AOL (America Online) 1992 - 1998

Staff Engineer, Senior Engineer

Developed network management systems software for NSFNET T3 and ANS networks. Developed ICMP and SNMP pollers for the ANS Network Operations Center. Developed alert management system. Developed trouble ticketing system applications for 7x24 Network Operations Center using early release of Remedy trouble ticketing system. System administrator and developer for monitoring and other support hosts worldwide (what would today be called DevOps). Helped Cisco develop NetFlow version 5.

Awards

1999 NANOG (North American Network Operators Group) Distinguished Member
1997 NANOG (North American Network Operators Group) Distinguished Member
1997 ANS President's Award
1996 ANS President's Award

Technologies

Programming	C++, C, Objective-C, assembly, flex/lex, bison/yacc, php, Java, JSON, protobufs, Boost Test, Boost asio, CORBA, XML Schema, XML DOM, UNIX shells, Perl, SQL, Qt, Cocoa, Crypto++, javascript
Protocols	TCP/IP, DNS, BGP, ICMP, SNMP, SPI, CAN, RS-232
Operating Systems	FreeBSD, Linux, iOS, macOS, Solaris, Windows 7, Windows 10

Education

B.S. Electrical Engineering December 1991

- University of Michigan

Interests

distributed systems
: One of the driving factors in my move to automotive software was my interest in distributed real-time systems; many small pieces working together to provide higher-level functionality. A typical modern luxury vehicle has over 100 microcontrollers running software.

efficient computing
: Long before we had smartphones and IoT, I became interested in low-power computing, the race to sleep paradigm on modern CPUs, and efficient software. I've applied some of the automotive world's strategies for low power consumption in my personal projects. I continue to be interested in high-level abstractions that can be compiled to native or near-native code: modern C++, JIT compilers for various languages, etc.

Other Software Experience

This is a partial list of software I've developed for personal and professional use, outside of the workplace. This list is only intended to indicate additional skills and technologies I've used effectively.

libDwm
: Approximately 38,000 lines of C++ library code used in most of my C++ projects. Licensed (without charge) to some of my previous employers for use in commercial products. Long-lived project.

libCredence
: Used for encryption and authentication of clients and servers on my home network. Used by many of my applications: mcblock, mcpigdgo, mcrover, mcweather, mccurtain and others. Under the hood it's using libsodium for the cryptography bits, and boost ASIO for network stream abstraction.

mcrover
: Monitors hosts, routers and switches on my home network. Web services, DNS, mail, network storage, Internet connectivity, etc. Client/server, fully meshed between instances. There is a curses-based display client and a Qt5-based client. I run the latter on my desktop as well as a dedicated Raspberry Pi with a dedicated display in my home office. Server and client run on macOS, Linux and FreeBSD.

mcblock
: Used to automatically block nefarious network traffic from the public Internet to my home. Uses libCredence for client/server access.

dwmgallery
: Replaced gallery3 on my web servers. Modern C++ using Wt. Fast and efficient, batch uploads, drag-and-drop uploads, captioning, editing, searching, authentication/authorization, etc. Runs gracefully on low-end hardware (Intel Atom with 4G RAM). No RDBMS: database is custom and small in comparison to even sqlite3 and supports ECMAScript regular expression searches.

mcpigdo
: project that combined my embedded skills with my long-time C++ and UNIX skills. Allows me to open and close and monitor my garage doors with my cell phone. FreeBSD on the Raspberry Pi, reading rotary encoders and magnetic switches and activating the garage door opener. Encrypted JSON-encoded status multicasted on my LAN, two-layer crypto used for commands. This was a full stack project, from device drivers for the rotary encoders to the web application that communicates with a server on the Pi over an encrypted TCP connection.

Tracks traffic to and from my web site at home. Uses packet sniffing to track packets and bytes as well as TCP round-trip times from SYN and SYN/ACK pairs. I have been collecting data since April 2011. Wt-based interactive charts provide a view of the data on my web site.

Mib++

Complete SNMP MIB compiler for SMIv2 written from scratch using flex, bison and C++. Instead of generating code, my compiler generates a database that can be queried from applications. Database lookups are speedy, with a reasonably small memory footprint.

Dns

A C++ class library for asynchronous DNS lookups. This was initially motivated by the need to perform millions of reverse (in-addr) lookups from the IP addresses in `skitter` and other network measurement data. Most record types in common use are supported: A, AAAA, CNAME, PTR, MX, NS, SOA, HINFO, MB, MR, MG, MINFO, LOC, SRV, TXT and CAA records.

sitesearch

The indexing and searching facilities used on my web sites. Indexes HTML pages, php pages, Wordpress blogs and gallery3 photo galleries. Makes them all quickly searchable via the search box on my web sites. Back end is C++ with flex lexers and bison parsers. Front end is mostly javascript. Data store uses Xapian.

mcback

A wrapper around dump(8) that backs up all of my FreeBSD hosts over the network on a nightly basis.

avrslave

Allows a PC to command various hardware actions on an Atmel AVR microcontroller via an RS232 connection: general purpose I/O, PWM, ADC reads. Includes code for both the AVR and the PC. Useful for hardware end-of-line tests of PCBs with Atmel AVR microcontrollers. The RS232 can use UART hardware or a software UART on any digital I/O pin on the AVR microcontroller. Microcontroller code is C++, PC code is C++ and C#.

miscellaneous embedded work

I've designed and deployed a small number of Atmel AVR based projects in my own vehicles using the Atmel AVR 8-bit microcontrollers. Examples: a simple controller to change the drive-by-wire throttle mapping, a controller to enable and disable different levels of dynamic stability control (and remember my settings across drive cycles), and a controller to flash my third brake lamp based on rate of deceleration (using the speed signal from the rear differential), LED cabin lighting with PWM dimming and separate red and white lighting.

phlegmp3

An mp3 jukebox system. It multicasts mp3 data over RTP, and uses CORBA for control, upload/download, etc. Nearly all of it is written in C++, though there is a java client to control the jukebox. The main client uses Qt for the GUI.

sparkle

Monitored and controlled X-10 devices in my home. Speaks to a few different X-10 devices (MR261, PowerLink, CM17A). A central servant provided a CORBA interface for clients.

Hobbies

software

Writing software is not just my profession; I enjoy it as a primary hobby. I always have personal software projects in progress, they provide for effective learning in my spare time.

system administration

I've been running some flavor of UNIX server at home for over 30 years. Today I have many FreeBSD and Linux hosts in my home that run 7x24, from Raspberry Pi systems to workstations and servers for software development, storage, web, firewall and local email.

automobiles

I've been fascinated with automobiles since childhood. I enjoy working on my cars and attending car shows.