DANIEL HUFFMAN
240.316.9908
Daniel.Huffman@gmail.com
16501 Shady Grove Rd #10672
Gaithersburg, MD 20898

SUMMARY
Software specialist with twenty years of experience in the areas of development, design and analysis of systems. A professional who has expertise including C++, C, Java, SQL, UNIX, TCP/IP Sockets. Dual expertise including Client Server distributed systems, server level and middleware development; as well as recent and legacy expertise in various embedded systems.

PRIMARY SKILL SET
C++, C, Java, SQL, UNIX, Threads, Sockets, Distributed Systems, UNIX shell scripts

PERIPHERAL SKILL SET
Bash, Git, TCP/IP, SSL, TLS, XML, ODBC, JQuery, JDBC, Hibernate, Spring, Rational Rose, Clear Case, J2EE, JAVAScript, UML, Fortran, Perl, Embedded C++/C, Assembly for Z80, 8086, IBM 360, RS/6000, AIX, Sun Solaris, HP-UX, BSD, Linux, DOS Lex, Yacc, Pascal, Lisp, Oracle, Sybase, Informix, DB2, Visual C++, .NET, CGI

CODE SAMPLES
https://GitHub.com/RattleTheCages

EXPERIENCE
Capital One (contract), McLean, VA
Software Engineer, August 2018 to February 2019

Resurrected mothballed DLL, debugged for new platform, and created new code base in git. This financial calculating DLL was in use at the time, but had not been worked on or compiled for years. The DLL processed information from several databases, providing predictions for "balance attrition model" originally written by "quantitative analysis people." Using C++, and MS Studio, compiled, debugged, and created production DLL for use in Citrix environment. Added multi-threading and a thread pool design pattern object. (Added other useful design pattern like singleton, of which this code was lacking. Provided documentation, in the code, describing them and their use.) Migrated DLL for use in new platform of linux in AWS. Created Python interface into DLL for other python developers in AWS environment.

USTech contracted to Raytheon, College Park, MD
Software Engineer, June 2017 to February 2018

Using Java, C++, worked on the SOTC team, supporting the JPSS-1 weather satellite, that was launched in November of 2017. Wrote and debugged code for telemetry and command and control interface. Fixed code, and wrote new code, in Java, C++, Python, shell script languages. Often interpreting, and referring to older revisions, in C++ and Fortran, to fix bugs in the newer version in Java, and Python. Using Clearquest bug reports, fixed bugs in the code baselines. ClearCase was used as version control. Debugging telemetry included TCP/IP and proprietary protocols, using sockets in C++ code.

NTT Data contracted to Raytheon contracted to National Weather Service, Silver Spring, MD
Software Engineer, February 2013 to April 2016

Worked on the National Weather Service AWIPS 2 platform. Debug code in Java using Eclipse and plugins. Translate C/C++ code to Java code from AWIPS one to AWIPS 2 attempting to allow the user experience be identical except for the enhanced graphics abilities of AWIPS 2. Developed new Java code from requirements or

without requirements by emulating AWIPS one. Debugged and wrote new Java code from code that was written by a different group of developers. Handled, debugged, and coded, in Java, resolutions to discrepancy reports created from trouble tickets from Weather Forecast Offices. When the front line tier one, and tier two could not fix the problem the Trouble Ticket morphed into a Discrepancy Report and handled by our group to debug. On AWIPS 2 platform: Cave UI, D2D, Radar Server subscription and heartbeat, Edex dx1, dx2, dx3, dx4, HDF5 files. Sockets using TCP/IP as well as pipes, the database, and nfs files facilitated communications between physical machines listed above.
Technologies include: Git revision control, PostgreSQL database, XML Java, including framework Spring, Hibernate framework, C++, Eclipse.

ACS, Germantown MD
Software Engineer, October 2010-July 2012
Migrate, improve, and add functionality to a database vehicle toll, traffic and financial system. Create new development to interface with existing system containing limited documentation. Interpret classical sequential C code to create modular and reusable C++ code for new development. The new development contains comprehension towards the ideas of retroactively replacing the 15+ year old C code with the new modular code, and the idea that the new code may be active and require to be maintainable 15+ years hence. Some aspects to achieve 15+ year maintenance on the new development are to use an extremely limited set of system calls and resources, and any system calls or resources are abstracted. Also an extremely limited set of C header files are utilized. The purpose being that as these header files change in the future, since they are not used by this code, no maintenance. will be required. E. g. the few required templates, e.g. a linked-list, have been developed so that the C++ standard template library is not required and therefore with new releases of that massive library, this code will require no maintenance. Interfacing into the database is achieved by ProC and SQL. An endeavor and enhancement I proposed was to increase database efficiency by the use of stored procedures, but their use was not agreed upon in this development cycle. The creation of abstracted database objects will allow for an efficient change to stored procedures when approved.
Tasked to troubleshoot and integrate multiple technologies, including, but not limited to, hardware, PL/SQL, Oracle SQL embedded procedures, C, C++, Java, Unix shell scripts in order to function seamlessly. Applications tasked to account for, and reconciliation of, vehicles traveling through toll lanes at highway speed using video license plate recognition technology. The application, which cannot error, do to ridged accounting rules, operates for one of the busiest toll systems in the world; namely the MTA in New York City. Data transfer from toll plazas to the main application was facilitated by using C/C++ UNIX sockets using the TCP/IP protocol.

Northrop Grumman, Sykesville MD
Software Engineer, February 2009-May 2010
Migrate, improve, and add functionality to embedded software written in C++. Write documentation about the past development environments for migration to future projects, processes, instructions for use. Update written documentation including the API for the libraries.
The migration was from an older PowerPC to the newer PowerPC, MPC7410, on a proprietary DoD processing board. The processing board and the software drivers control low level monitoring and mechanics of nuclear reactor power plants. Being that it is extremely important for the software and drivers to perform correctly, code migration includes inspection of the assembly code and assembly code level proof of proper functioning. Unlike the boards that are being replaced. The newer PowerPC chips pipeline, so examination of the assembly code must insure that I/O functions in the correct order and out of order I/O does not occur.
The migration involved resurrecting code and a development environment that was obsolete because the customer deliverables included the old code libraries, the new code libraries, and proof that the new code works identically to the old code. The obsolete development environment was Microsoft Visual C++ 6.0 with Nucleus EDE plug in, and the Diab 4.3g compiler. The new development environment is Wind River Work Bench 2.5, using the Wind River 5.5.1.0 compiler.
Create new and perform existing unit tests on code using Parasoft C++ Test tool. Create documented test results on modified and existing code.
Write procedure documents for technical audience on update procedures to military grade touch screen apparatus.
Follow and improve documentation on test procedures for a turbine generator control and fail-safe system.
Insufficient documentation required me to draw on electronic as well as software expertise in order to clarify,

improve, and proceed with the testing of the embedded system wind turbine apparatus. For example knowledge of a circuit board, its grounds and voltages, was required to force faults testing if the code recognized and logged the fault.
Design, implement, and test a post processing utility that took machine language output, used in a DSP, from the Code Composer compiler and convert the raw hex into a usable format for use by Corelis ScanExpress 1.05 flash memory burner connected to a high speed USB JTAG controller. The utility also created and included a military required code checksum, and flash memory offsets. Since this implementation of the Code Composer 3.1 executed in Windows, I used the Microsoft Studio 2005 IDE to implement this command line driven post processing utility. The utility was configurable, on the command line, for use in the post processing of many different DSP applications, for the Rod Drive Power Supply system used in the nuclear power plant on Navy Aircraft Carrier ships. Implemented the utility by including it in the Code Composer project files of the many DSP applications requiring its execution during the post processing phase. Tested results by flashing the memories of the DSPs, connecting them to the system, and executing various tests. Used Code Composer IDE and emulator to modify, write, and debug code on both the simulator and loaded onto the DSP from the Emulator. Developed, debugged, and tested Java Front-end to nuclear power plant on Navy Aircraft Carrier ships. The Java front-end is a human-computer interface providing nuclear control rod operation and status and provides controls for operation augmentation.

Kastle, Alexandria VA
Software Engineer, September 2008-November 2008
Design, develop and deploy embedded software for a dual redundant linux single board computers. The project involved using their proprietary current library of code to create an embedded driver that maintains a connection from a building's fire safety panels to Kastle's internal database and response centers. This project used C/C++, the gnu compiler, and a proprietary development environment. Short project assignment.

Raytheon, Landover MD
Software Engineer, April 2007-July 2008
Troubleshoot, debug and perform maintenance on the NASA EMD, Earth Observatory System, the deployed and operational system code tree. Design and write new code for development future deployed system code tree. Troubleshoot and debug non-conformance reports for the Order Manager Server subsystem, the most complex subsystem of the total system. Includes manipulating all tiers of Order Manager: the database, the C++ middleware server, caching, and the customer GUI. Bugs include segmentation faults, memory leaks, stack overflow, etc. Working closely with team members on reengineer to simplify and reduce lines of code, integrate new C++ threaded frameworks, review team member’s code. Debug, maintain, and create new Sybase stored procedures. Debug, maintain, and create new C++ code for system functioning. Debug, and maintain Perl and CGI code. Tools used in this environment include: g++ Gnu compiler, vi, gdb, Gnu Debugger, Purify, Valgrind, Clear Case version management, RougeWave code library. Other duties include work on the Datapool subsystem, when problems spanned both subsystems. Using NetBeans IDE, Debugging Java and JavaScript. Debug Java jdbc code, using J2EE. Reviewing and verifying other developer’s Java code. Update stored procedures to work with both C++ odbc and Java jdbc.

TCOM, Columbia MD
Software Engineer, January 2007-March 2007
Updating embedded legacy microprocessor software to embedded Linux software. For use in TCOM’s primary product, aerostat, a dirigible that is used by US and foreign military as radar, inferred, and other intelligence gathering platform. Embedded software controls the flight guidance and control mechanisms as well as reports telemetry for on board systems.

ACS, Germantown MD
Lead Software Developer, March 2006-November 2006
Lead programmer for development of new generation of tollbooth lane collector software. When deployed, soon, this software will be responsible for tracking millions dollars worth of tolls each month. The lane controller is to

store 3 months worth of data if the network to the database is non-functional and thus, the lane controller is to work autonomously, if necessary. The software is written in C/C++, UML, using the Rational Rose developer's environment, using Rational Rose realtime version, and compiled to a linux platform that is configured for real-time processing. Server communication software is written in Java using J2EE and Java servlets. Code revisions are stored in Clear Case, and task tracking in Clearquest. The lane controller operates several devices, including, but not limited to, a traffic light, entry gates, vehicle detection induction loops, vehicle detecting lasers, toll collector's touch screen GUIs, speed radar device, violation enforcement digital cameras (to be triggered when the vehicle performs some violation like not paying toll). Duties included writing and unit testing code, writing and proofreading design and other documentation to be presented to the customer, organizing the entire system for functionality in the field, on-call (last tier) for problems.
As a lead software engineer, the team was 8 people, including myself. During my time we went through 2 complete life cycles, and some fractional cycles. First the manager, and I, met a few times discussing what he wanted to include and high level implementation, of the next set of requirements. The requirements were given to me to revise. After the requirements were good with both my manager and I, the requirements were given to me to implement. I arranged a meeting with my team to give the requirements, discuss them, and then divvy up the work to the team members. I would give a deadline for my team members to get their projects completed. During my teams work I would, informally, go talk to them to see how they are doing, and answer questions, and help with problems. My deadline was usually a 2 weeks, or more, before the deadline to complete the project and include it in the code base. (Things are a little different, now that Jenkins, and formal DevOps is available.) This would give me some time to integrate everyone's code into a working application. (And give some time for stragglers who I, encouraged, to finish, and perhaps some time to fix problems I found, if any.) I take the working application to the lab with the full working mockup of the apparatus. I tested the application on the apparatus to make sure the new functionality was working correctly in the real environment. Then I would submit everyone's code to the official codebase.

Secured Services, Baltimore MD
Software Developer, August 2005 to March 2006
Worked in a security product that allows secure single sign on into a diverse array of systems like web, unix, windows and legacy mainframes. Reconstructed a project that was in sever disarray by former employees and the fact that the code was a conglomeration of several different projects from several different companies that were merged in to one company. Compiled the code into a working product in linux and Solaris unix operating systems. The linux product was put into production at the customer site; bank IBC. The code consisted of C, JAVA, XML, pearl, and other technologies including J2EE. Many bugs had to be fixed along the path of fixing the compilation of the project. Completion was done by gnu C compiler, Sun Solaris compiler, and JAVA compiler. The code consisted of in house code and opensource code; such as DCE1.2, LDAP2.0, DB3.2, tcl/tk, openssl0.9.7, apache, xml2, zlib. Some of the opensource had been altered to include in house customization, some had been left pristine. The executables were policy and encryption servers, clients in command line and GUI form, and java on web pages. After reconstructing the product into a working distribution, reorganized the entire source tree into a much more cohesive architecture. Answered questions and found solutions about operating systems and coding for employees in other departments, such as quality assurance and management.

Telenix. Columbia MD
Software Engineer, Sept 2002 to December 2003
Ported web server application, Apache, to use in the proprietary Telenix network emulator. Port included caching. Designed protocol interface applications to expand Telenix's network emulator to include Signaling System Seven (SS7) telecommunication protocols ISUP, TCAP, and MTP. Wrote detailed design documents. Created proof of concept prototype drivers for designs.
Researched, documented, and created proof of concept applications using Secure Sockets Layer (SSL) or Transport Layer Security (TLS) protocol for using in the network emulator's TCP/IP environment. Specifically for, but not limited to, use in the DNSSEC emulations. The simulator application executed on Solaris, however I endeavor to write code portable over several UNIX platforms.

NAPA. (Consultant), Atlanta, GA
Software Engineer May 2002 to June 2002

Worked with team to port and extend existing RADS server software backend with DB2 to partner company in order to provide wireless hand-held bar-code scanning units for use in their auto parts distribution centers and auto parts stores. Code uses Visual C++ and C#, SQL, and operating system scripts. Wireless hand-held units, manufactured by Symbol, require C embedded coding.
Job duties included being the Lead programmer, Program Manager, System Administrator, and counselor to IT department. Managerial duties required the installation of a full life-cycle development environment, including development, test, and production. System administration duties included administrating DB2, Windows 2000 Server, and Microsoft Studio .NET. Lead Developer duties included porting and writing code in C, C++, Visual C++, C#, and embedded C. Lead programmer duties also included training of in house individuals to maintain and continue development on the system.

WorldCom. (Consultant), Colorado Springs, CO
Software Engineer, May 2001 to December 2001
Worked on team to architect n-tier system comprised of components database, middleware servers, front-end GUIs.
Consulted on performance and mutual exclusion issues in distributed system with many nodes and processes.
Guided team to make fresh decisions on how to distribute workload over this vast client server architecture.
Wrote documentation describing the n-tier system, n-tier system node function and cooperation, and system protocols. Wrote high-level overview, and low-level, detailed description documentation.
Designed, wrote, unit test, C++ object components for this n-tier, client server, system.
The system used SQL for Oracle, Sockets, and C++ on a DEC/Alpha system executing UNIX.
The system, Traffic Data Store and Real Time Monitor, reports, to web users, usage statistics for their toll free numbers.
Worked on a large team in a full life-cycle development environment.
C++ objects I designed and coded include ODBC, Sockets, Threads, Mutual Exclusion Locks, and some basic object functions like string and date objects. (Using STL was taboo in this project.)
Wrote design documentation as well as code.

Decision Consultants, Westminster CO
Software Engineer, September 2000 to December 2000
Wrote Java Servlets connected to a SQL server using JDBC for Outward Bound's classes' database.

EDS. (Consultant), Boulder CO
Software Engineer, April 2000 to August 2000
Wrote database queries in SQL for the Census 2000. Wrote C++ and C programs to compute call center statistics. Developed in C++ and C, using business rules and ODBC to verify and update data from various call center databases to a central database. Developed in JAVA and JDBC, stand alone programs, to verify and update various call centers to a central database. To satisfy Congressional inquiries quickly, wrote UNIX scripts and SQL queries to find citizen records, with limited, or incorrect (i.e. misspelled) information.

Information Mechanics Inc., Littleton CO
Software Engineer, July 1999 to March 2000
Reported to the Director of Infrastructure Development, and worked on system architecture and development for a collection of robust, high-performance, scalable servers. Participated in planning and implementation of specialized custom servers, application servers, and client interfaces.
Custom development included C++, multithreaded client-server architecture, and communications.
Teco energy project included a JAVA interface for dispatchers, and an HDML hand-held interface for drivers, technicians, and meter readers. Field information, such as meter readings or technician job completed, are updated in real-time, from the field, on the dispatcher's JAVA GUI, and the database. The back-end is a Sybase database.
Lafarge project included a JAVA GUI for truck dispatchers, and a HDML hand-held interface for the truck drivers. Truck information, such as loading, pouring, cleaning, are updated in real-time to the dispatcher's JAVA GUI and to the database. The back-end is an Oracle database.
Mobeo project. Optimized Mobeo code for RIM and Blackberry so that most of the other applications could fit in the memory concurrently. The Mobeo project lists, in real time, stock quotes on RIM pagers. (Now anyone can get this, but back then it was novel, expensive, and for large Wall Street players.)

IBM Corp. (Consultant), Boulder CO
Software Engineer, February 1998 to April 1999
Worked on the Transportation Routing & Information Pricing System (TRIPS) application that provides users of IBM a list of all current transportation contracts and the ability to select the most efficient carrier that can meet the service requirements of a particular shipment.
Designed and wrote from scratch a multi-threaded three tiered performance CGI web and information server capable of servicing 100's or request per minute in C++ for the RS/6000 platform designed for portability.
The server served as the middle and front-end tiers of a three-tier system. The back-end is a DB2 database.
Designed resource management libraries to control server resource allocation, including memory, and to ensure that if other processes on the host system become resource hogs, the execution of the server will not will effected.
Designed and implemented thread-safe C++ shared memory library including two binary search tree templates (one self-balancing), and a queue. This was done because the AIX system manual highlights what subroutines are not thread-safe. One of the library elements was the system hash table. (Furthermore, I can prove that with a large amount of random data being inputted into a hash table will take longer to search and sort than using a binary search tree.)
Designed and implemented a thread-safe encapsulation of ODBC (CLI). These C++ object take care of all the ODBC `handles' and leaves the programmer with a very simple database interface.

US West. (Consultant), Denver CO
Software Engineer, April 1997 to February 1998
Design, code and unit test new functionality added to US West's Call Handling project.
Designed, implemented, and coded a shared memory cache used singleton instance interfaces for each executing entity.
Worked with a small team to design, code, implement, and unit test new rules and call reasons for an existing server. The existing server is written in C, the new functionality was designed and coded in C++. The new functionality included new objects, SQL queries, and integration of new client/server interactions to previously unconnected and new clients and servers.
Assisted in design of new functionality that locates and transfers to operators via a user entered operator code.
Effectively worked as part of the large Call Handling team that included people working with: Oracle DBs, Conversant Voice Response Units, Sequent multiprocessor boxes, dynamic HTML output for Netscape, Windows NT boxes.

Atlantic Aerospace Electronics Corporation, Greenbelt MD
Software Engineer, August 1995 to April 1997
Program the framework in C++ with UNIX scripts.
Programming in C++, TCP/IP sockets.
Design and program an interoperability CAD framework using client/server and object oriented programming technologies. Demonstrate framework to clients, customers and interested parties. Design the architecture of the framework. Included proprietary codes CAFÉ for electromagnetic analysis. A GUI controlled execution flow, while the framework transferred data and formatted data into formats each COTS or proprietary application needs. The framework allowed multiple passes into any application; the execution flow is up to the users. The framework adhered to object oriented programming theory. Tasks were delegated to C++ objects.
This allowed for all the benefits of object oriented programming, such as object reuse: ported two objects to DOS for use in a program that formatted analysis information from a microwave tester; modularity, objects can be replaced for changed without changing other objects; data hiding, objects do not allow direct access to their data.
Networking is achieved by objects that transfer other objects in TCP/IP programmed directly on top of sockets.
Porting to Windows NT will be achieved by porting all the C++ objects except the networking object. Only a few objects need maintenance in the port because of object oriented programming.
Write concurrent programs for a proprietary multiple processor system (named BLAZER). The system had 16, 32 or more processors, and PCI bus. The threads communicated through shared memory; and communicated with the command program (on a sparc 4) via sockets. Integrate Python algorithms and user interfaces into existing framework Python was written by graduated students from Carnegie Mellon University, but required bug fixes and revisions to incorporate into the system,

University of Maryland at College Park, College Park MD

January 1993 to May 1995
Software Engineer
Design, code, and maintain programs involving Client/Server with sockets and SQL database.
Write programs and work with other campus organizations to create and maintain data transfers between a proprietary SQL database (Griffen) and an IBM mainframe using IPX, TCP/IP and SNA for automation of operations. Write programs to connect UNIX and WINDOWS (between and across the two systems) using client/server TCP/IP paradigm and the internet.
Write and modify a menu system and other Novell network system programs (using Novell's C interface). Write programs for UNIX, WINDOWS, DOS, using C, C++ (emphasizing Object Oriented programming and reusable libraries), Pascal, SQL. Write Dynamic Link Libraries (DLL) for Windows. Write Quattro-Pro spreadsheet. (The computer systems included a 110 PC network, 2 Novell file servers, IBM RISC/600, Griffin Series/5, IBM 9370. The network consists of 10baseT, Thinnet Ethernet as well as a FDDI and a Token-ring network.

University of Maryland at College Park, College Park MD
Network Technician, August 1992 to January 1993
Maintain the BSOS Novell network. Troubleshoot and repair network, software and hardware problems. Install computers, print servers and other items to the network.

Distribution Plus, Frederick MD
Electronic/Computer/Network Technician, July 1989 to October 1990
Install UNIX, install Novell networks, repair computer equipment. Troubleshoot using Oscilloscopes, Logic Probes, Multimeters, Equipment Calibrators, circuit analysis and technical expertise. Isolate components in error. Repair items include: 8086, 286, 386, 486 systemboards, disk and video controllers, monitors, dot-matrix and laser printers, hard-drives, CD-ROM readers and other peripherals. Some pieces required component level repairs. Consult staff and customers with regards to technical, hardware, software, network problems or education issues. Run test/benchmark experiments with varied UNIX and Novell versions against a variety of systems and components. Recommend modifications to increase reliability and performance. Perform programming in BASIC. Write test and work procedures. Write services reports.

Montgomery College Physic Lab, Rockville MD
Lab Technician, August 1987 to July 1988
Follow written and oral instructions, prepare, test, and remove lab experiments and equipment for electronics, physics, and astronomy classes.
Tests require electronic, physics, waveform, etc, computations for analysis. Equipment included: oscilloscopes, Logic Probes, Multimeters, Frequency Generators, Frequency Counters, Variable Power Supplies, Computers, Resistance Boxes, Geiger Counters, Nuclear Sample Prepares, lasers, etc. Using schematics, assemble, solder, test and troubleshoot PCBs.

EDUCATION:
Bachelor of Science in Computer Science. University of Maryland at College Park, 1995.
Associates of Arts in Electronic Technology, Electronic Technology, Montgomery College, Rockville, Maryland, 1991.
Graduate course, C programming, MIS department, University of Denver, 1991.
Graduate courses, Computer Security, High-Speed Networking, Johns Hopkins University, Maryland, 1997.