MARK B. ROSENTHAL **781-648-4031**

SUMMARY OF QUALIFICATIONS

Extensive experience designing and developing mission critical software.
Experience at all levels of software development from end-user applications to language development tools and OS internals.
Experienced in gathering requirements from non-technical people and translating their input into functional software.

PATENTS

Automated Association of Operations with Objects in a Graphical Window - Rosenthal, Turok, and Abramovich, #6,750,882.
Processing Technique for Digital Speckle Photogrammetry - Stetson and Rosenthal, pending: #10/409713.

COMPUTING ENVIRONMENTS

Languages:	PHP, Javascript, Java, C++, C, Perl, Lisp, VRML, assembler
GUI:	Swing, AWT, X Window System: Motif, Xt Intrinsics, Xlib
Web Technology:	HTML, CSS, AJAX, XML, SOAP, Dreamweaver, WordPress
Operating Systems:	Linux, Solaris, AIX, other Unixes, Windows XP, Windows ME
Database:	MySql, Informix
Enterprise Technology:	JSP, Servlets, XML, Java Beans, Corba IDL
Other Tools:	Make, GNU Make, Shell-script (Bourne, Csh, Korn, Bash), Emacs, Clearcase, etc.

TYPES OF APPLICATIONS DEVELOPED

- Web Applications
- Language Development Tools
- Network Management
- Graphical User Interface
- Word processor
- Device control

WORK EXPERIENCE

Arlington Software Enterprises, Arlington, MA **2002-present**
Independent software consulting practice. Clients have included:

Accordare, Inc. – Implemented **Java** classes to automate fetching of client's data from webservers where client has accounts. For those servers that implement a web service API, data is fetched with **SOAP** requests. For those servers that do not, the code sends HTTP POST or GET requests, parses the HTML response, and extracts and stores the information the client desires.

Secure Access Research, LLC - Development of a new building access control system. Code written in **PHP**, **MySQL**, **Javascript** and **C++**. GUI designed using **Dreamweaver** and raw **HTML**. Developed generic and highly configurable core code to generate data input screens and create database records from a common configuration file. Wrote **extension to the PHP engine** in **C++** to allow PHP applications to access encryption hardware. Developed custom GUI components using **AJAX**. Also wrote an SQL parser in **PHP** to allow the code to recognize an augmented SQL syntax. Advanced users writing their own SQL queries can use the augmented syntax to cause input fields to be displayed in the user's browser, and then insert the user's responses into the query. Designed protocol for communication between centrally located server and processor located at door, and implemented remote-procedure-call mechanism in **PHP** that passes data from client to Apache server via HTTP POST request.

Karl Stetson Associates, LLC: - Development work on new mechanism for non-destructive materials stress testing. This application captures images of laser interference patterns from a video camera and compares them via a proprietary algorithm implemented in **C++**, and generates viewable results in TIFF format.

LTX Corp.: Automated analysis of similarities among client-submitted stack traces to prioritize bug reports so the team could decide which bugs to focus on to get the greatest improvement in mean-time-between-failure of client's chip testers. Analysis code written in **Perl**.

JKI, Inc. – Reimplemented this company's web newspaper with **WordPress**, and implemented custom functionality by writing WordPress plugins in **PHP** where possible or modifying the core WordPress engine in cases where the changes could not be accomplished via a plugin. Some of the custom functionality implemented included: 1) code to import an RSS feed into the WordPress database, including data extracted by parsing the HTML pointed to by each article's link, 2) major modifications to WordPress' template selection system, 3) rewrote code for WordPress' user administration page to optimize database queries, so a system with 2,000 users takes 5 seconds to display the page rather than the 4 minutes required by the previous code. Also configured and administered **Linux**/**Apache**/**MySQL**/**PHP**/**WordPress** system. Solved problem of some outside domains rejecting our email, partly by software debugging techniques and partly by acting as liaison between system administrators from different webhosting companies.

Verizon Communications (formerly GTE Laboratories), Waltham, MA **1992-2001**

Principal Member Technical Staff – responsible for software used by GTE/Verizon to monitor its nationwide network of telephone switches.

Project leader for NeMoW (Network Management on the Web) SS7. NeMoW is used by Verizon Telephone Operations to manage its network of telephone switches throughout the U.S. The SS7 Project delivered enhancements to NeMoW to support Verizon Telephone Operations in monitoring SS7 switches in Verizon's network. Coding done using **Java**, **Swing**, **CORBA, C++, Informix**.

Designed and wrote object oriented infrastructure code for the NeMoW client. Designed base classes to facilitate code reuse. Implemented in **Java**, **AWT**, and **Swing**.

Designed and implemented NeMoW dashboard, an overview of the condition of the telephone network. Implemented in **JSP** using **Tomcat** server. The dashboard **JSP** code uses **Java beans** to communicate to services provided by the NeMoW back end. This includes a login page which talks to NeMoW's password verification service. After the user logs in, the display fetches data from other NeMoW services. This data is aggregated according to user-specified criteria, e.g. by customer, by type of phone switch, by physical location of switch, etc. Displayed data is updated at regular user-specified intervals. User can click on any column to sort aggregated data as desired, or can click on any individual piece of data to drill down into the details of that data.

Invented a patented solution to a common problem with menu-based user interfaces. Menus are often created from a hard-coded list that must be manually updated for each new menu item. A programmer adding a new item can easily miss adding it to some menus. If the menu item applies to an object that appears in different types of windows, customers find they can use the menu item from some windows that display the object, but not others. They complain that if a window displays an object, its menu should contain all operations that apply to the object. My solution has each menu item advertise the functionality it requires of a window, and each window advertise the functionality it offers. A menu item is included in the window's menu only if the window offers all the required functionality. In **Java**, "functionality" mapped nicely into "interface", but the concept can be implemented in any language.

Wrote **VRML** and **Javascript** portion of NeMoW's 3D network map and wrote interface to allow code to be called from the Java portion of the NeMoW code.

Designed and implemented Unix shell scripts (**ksh, bash, perl)** for automated nightly build of NeMoW, including automated installation of newly built NeMoW across network, automated invocation of a suite of regression tests, and automated e-mailing of a summary of results to affected developers.

Designed and wrote major portions of the user interface for TONICS (Telephone Operations Network Information Control System), the predecessor of NeMoW used by Verizon Telephone Operations to manage its network of telephone switches throughout the U.S. Implemented **C++** base class hierarchy to facilitate code sharing among different TONICS windows.

Wrote highly optimized, low-level **Xlib** code for the TONICS alarm window to allow it to keep up with a rapid rate of incoming alarms. I designed the code to eliminate round-trip X protocol delays, minimize the number of pixels the X server needs to redraw for each new alarm, and to issue those X protocol requests that the server can satisfy fastest. I also did major work on windows which provide a variety of views of telephone switches and their alarmed state, including a network map, a geographic netmap, and a tabular display. Also filter and view editor windows. Coding was done in **C++**, **Motif**, **Xt**, and **Xlib**.

Contract Programmer, various clients **1990-1992**

GUI: Securities tracking system in **Motif**, conversion from Athena to Motif widget set, image processing in **Motif** and **Xlib**

Standards: Converted libc to comply with X/OPEN standard.

Adaptive Optics Assoc., Cambridge, Mass. **1985-1990**
Senior Programmer

Wrote microcode assembler that supports user-defined instruction set

Wrote relocating linker, loader, and support routines in **C**.

Wrote daemon and user interface to drive devices controlling laser-optics experimentation.

Developed standards for **C** programs, program build (**make**), and revision control (**SCCS & RCS**) usage, and trained programmers in adhering to the standards.

Installed devices; wrote **C** code to support them; diagnosed **RS-232** communication problems; designed cables.

Wrote numerous programs for displaying experimental data using the **X Window System**.

Fortune Systems Corp., San Carlos, Calif. **1981-1984**
Senior Programmer

Reverse engineered Wang word processor in **C** under 4.1BSD **Unix**.

Interfaced printer at both hardware and software levels.

Designed printer driver code which realizes a document on diverse printer hardware while interfacing to higher level software through a common API.

Medical Information Technology, Cambridge, Mass. **1977-1978**
Software Engineer

Developed high-level language for creating data base applications to run under **MUMPS**. Since MUMPS was the only available language on the system, the compiler generated MUMPS code and was itself written in MUMPS.

Developed a high level database query language which generated MUMPS code.

Digital Equipment Corporation, Maynard, Mass. **1972-1977**
Software Engineer, Languages and Small Systems Group

Developed DEC's popular EDU20, EDU25, and Multi-User BASIC RT-11. These were integrated BASIC interpreters and multi-user OS's. The EDU series ran on PDP-8 hardware, and MU-BASIC ran on PDP-11 hardware. In an era when computer memory was very expensive, the EDU20/25 product supported up to 8 simultaneous users running BASIC programs in a mere 12 Kilobytes of memory. Development was all hand-coded machine language. Code written included language parser and runtime execution phase, floating point simulation package, device drivers, task scheduler, and other OS internals.

Oberlin College, Oberlin, Ohio **1969-1972**
Programmer

Developed a comprehensive debugging package for the IBM 360/44.

EDUCATION

Indiana University, Bloomington, Indiana: Linguistics, 1979
Oberlin College, Oberlin, Ohio: B.A. in Physics, 1972

REFERENCES

Available upon request.