Software experience highlights (dam-soft)

David Morano
david.a.morano@gmail.com
1(781) 388-1799

Some selected software experience is highlighted with these outline notes.

== lower-level software architecture and design

+ designed and developed much low-level code in the embedded space:
- power-up diagnostic code
- peripheral device discovery and initialization
- other processor discovery (for multiprocessors) and initial synchronization
- system initialization code (memory, MMU, processor traps and interrupts, et cetera)
- coded for both single processors and symmetric multiprocessors
- boot-up code (boot from both ROM or external sources)
- object file formats read were 'a.out' and 'COFF'

+ architected and designed a real-time-OS (later called 'DAMOS') for use with our embedded computer systems:

- modeled mostly after DEC VMS (or DEC RXS-11)
- event flag driven for user space asynchrony
- I/O API from user-space was something of a cross between standard UNIX I/O and VMS I/O
- all I/O from user-space is asynchronous; wait on an event flag for synchronous behavior
- flexible user-space process interrupts (like UNIX signals, but more closely modeled after DEC VMS Asynchronous Signal Traps (ASTs)
- interruptible kernel
- separate I/O kernel with special I/O-kernel threads (called "forks" after the use of that name for the corresponding entities in VMS) that can be prioritized and preempt lower priorities
- user-space process preempting priorities
- kernel and I/O-kernel were architected and designed for an especially high degree of kernel-I/O thread parallelism and preemption (similar to the architecture of DEC VMS)
- later variations of this OS lowered the degree of preemption possible in the I/O kernel exchange for faster overall OS and I/O throughput
- supported asynchronous process-termination waiting in a fashion similar to making an I/O request

+ designed several device drivers for DAMOS above
- serial line
- network packet interfaces
- bus-window interfaces to other processors in the overall system
- custom (intelligent) data-processing hardware
- standard packet (Ethernet) interface hardware (AMD LANCE)

+ architected and designed a ROM boot-up and program diagnostic monitor, called eXended Debugging Trace (XDT) monitor:
- some built-in diagnostic capabilities (usually light tests of included hardware and peripherals)
- supported examination and manipulation of target code
- supported break-pointing and tracing of target code
- supported disassembly of target code
- supported various boot-up mechanisms

+ enhanced an existing RTOS, "Tasking Operating System" (TOS), to support almost the same OS API as DAMOS above (named Extended tasking

Operating System (ETOS)

+ designed several device drivers for ETOS (an RTOS)

+ architected and designed diagnostic tool software (run on the central processor in a functionally partitioned multiprocessor) for use in testing other processor components; this tool substituted for what would normally be production device drivers running on the same main processor; ran on the enhanced RTOS above

+ participated in the development of diagnostic software (similar to above in some respects) but integrated with other software for system testing of hardware and embedded software; this ran on the 'TOS' (non-enhanced) RTOS

== medium-level software architecture and design

+ designed embedded software for processing various networking protocols
  - AT&T custom protocols (used for both interprocessor communication and connection management signaling)
  - LAPD (both for interprocessor communication, and for ISDN layer-2 signaling)
  - X.25 for multiple purposes including standardized ISDN data transport; ran on top of LAPD links
  - all ran on DAMOS and were multithreaded
  - supported thousands of simultaneous layer-2 links and layer-3 channels
  - much code deals with packet buffer management (from higher level and down to hardware)

+ architected and designed interprocessor communication software for intelligent I/O purposes
  - system device tree population (part of synchronization to next higher level)
  - boot and reboot code (as directed from higher level software)
  - I/O message request-response communication code
  - coded slightly different incarnations for different jobs

== CAD software (all on UNIX)

+ many small enhancements to the user interface to add short-cuts for commonly used tasks: renaming components, renumbering or renaming sets of components; these codes often involved both user-interface handling (graphical input and output) as well as database lookup traversals and appropriate updates

+ design and development of a general mechanism (involving both the user interface and the back-end circuit databases) for supporting very flexible hierarchical circuit jumping across both a single design as well as a set of designs normally thought to be separate and distinct; jumping is even possible between the overall "frame" (system) design, circuit board designs, and integrated circuit designs

+ design of a new facility to tag signals (signal nets) with switching-speed characteristics; this involved both user interface work (the user tagging commands) as well as the back-end database to store the characteristics; additional software would then extract the results of a 2% dimension EM-field simulation and match those results up with the original signal data (previously stored) to find those signals that violated specified cross-talk thresholds

+ circuit library software (circuits are represented in SPICE or ADVICE language)
  - translation of flat circuit code (ADVICE) modules into library modules (required parsing of ADVICE language)
  - extraction of circuit modules by name

+ software to create (compile) circuits in ADVICE (SPICE) language
	- circuits to simulate distributed signal transport (for integrated circuit design)
	- simple circuits for inversion and complimentary signal generation
	- circuits to simulate IC carrier environments
	- all parameterized (designed permittivity, permeability, and impedance)

+ language or data translation
	- custom HDL to ABEL (a pseudo-standardized vendor language)
	- FPGA native representation to factory standard
	- SPICE <=> ADVICE
	- various object file code format translations:
		› a.out to SREC
		› COFF to SREC
		› COFF to Intel-Hex
		› SREC to Intel-Hex

== automated test research software

+ architected and designed software to build automated test suites for one or more specified test categories (AUTORUN)
	- test suites are built from a database of tests
	- tests are written in a separate language (Prairie; resembles English)
	- the software then conducts the tests by controlling an automated test system framework (ask for more details)

+ evaluated artificially-intelligent software to perform automated tests
	- reads English-language test plans (meant for human reading)
	- tries to "understand" what it reads and to
	- write a test program in the Prairie language (which itself resembles English) to test the system software through event stimulation

== cell phone research on future phone designs

+ miniaturization of the overall hardware and range of radio modes (CDMA, TDMA, analog, and GPS) dominated our priorities for future phone design; power consumption (low) was still a very high priority

+ responsible for research and evaluation of future cell phone designs, with emphasis on the radio design and the computer system design

+ performed software architecture work to map our existing code base (both general control code and digital-signal-processing code) to a new computer-DSP system

+ the main goal with future computer systems was to minimize the total hardware devoted towards computing of all types (primarily the general control code and the DSP code)

+ evaluated how to port our embedded OS to new processor architectures

+ analyzed our existing DSP code base for possible porting to C-language code running on a general purpose (not a special DSP) processor; and visa-verse, analyzed our existing general code base for possible porting to a DSP

== simulation software (all on UNIX)

+ distributed circuit simulation software
	- creates and schedules circuit simulation tasks

- tasks identified through enumeration of circuit and environmental parameter lists
- automatically (at task boundaries) load balance
- built on a load-balancing remote execution framework

+ machine microarchitecture simulation (SimpleSim)
- performed simple simulated execution of MIPS machine architecture on SGI-UNIX
- the sophistication of the simulated machine was similar to SimpleSim (industry standard?)
- was primarily used for program behavioral analysis
  › determination of Hammock branch constructs (stored in a constant database)
  › static and dynamic program branch behavior and characteristics
  › program dynamic subroutine frequencies
  instruction operand (register and memory) dependency tracking and analysis
  › subroutine tracing and coverage
- was primarily object oriented with the exception of the simulated execution of instructions
- featured run-time loadable evaluation (plugin) objects (like for various branch predictors)
- most of the entire simulated OS was emulated at the OS API interface (used a very fast database to identify OS API entry points)
- target program is mapped (as w/ UNIX) from 'ELF' program object-files

+ machine microarchitecture simulation (LevoSim)
- performed simulated execution of an elaborately hierarchical set of machine microarchitectural components organized for extracting maximal instruction-level-parallelism
- all simulated machine components are object oriented -- each component makes up a software object with its sub-components being component software objects
- the simulated execution of instructions was not object oriented
- most of the entire simulated OS was emulated at the OS API interface (used a very fast database to identify OS API entry points)
- simulated MIPS architecture on SGI-UNIX
- target program is mapped (as w/ UNIX) from 'ELF' program object-files

+ participated in the design and development of a trace-oriented program simulator (FastLevo)
- this was not object oriented
- but this simulator was used for the majority of the data for the first part of my doctoral research work
- simulated MIPS architecture
- OS calls are transparent to simulation

+ machine microarchitecture simulation (OptiFlow)
- performed simulated execution of a moderately hierarchical set of machine microarchitectural components organized for extracting high instruction-level-parallelism
- all simulated machine components are object oriented -- each component makes up a software object with its sub-components being component software objects
- this simulator was used for the majority of the second half of my doctoral research
- this simulator featured run-time loadable evaluation (plugin) objects (like for various branch predictors)
- simulated Alpha instruction-set-architecture for DEC True-64 OS
- loaded from 'ECOFF' program object-files

+ program trace conversion and analysis
- machine objext code to custom binary trace format

        - comparison of traces (common trace format and management)
        - dynamic loadable (plugin) architecture
        - programs:
                › leveosim (generation component)
                › simplesim (generation component)
                › dammint
                › dbcvout
                › stripopgarb
                › tracedbx
                › pixie2levo
                › tracepixie
                › tracecopy
                › tracecmp
                › icount
                › fcount
                › tracestat
                › traceproc (pluggable)

+ XML
                › levosim (generation)

== UNIX library software

+ various ulility objects (containers, other)
        - queues and FIFOs (various degrees of atomic insertion-deletion)
        - vectors and lists
        - hash-table types (general and strings)
        - "maps" (ex: string-integer, visa-versa, et cetera)
        - general text indexing

+ much string manipulation software subroutines (mostly non-object)
        - safe general counted strings
        - path-name and file-name manipulation

+ various marshaling objects (sort of like 'XDR' by Sun Microsystems)
        - serialization and deserialization
        - general buffer management and manipulation

+ low-level (non-object) machine-independent data-representation code
        - network-order (big-endian; like 'XDR')
        - little-endian

+ UNIX file-system directory traversal objects
        - wdt
        - fsdirtree

+ message queue objects
        - for POSIX (using POSIX message queues underneath)
        - for storage in files (using record locking for atomic access)

+ various mail-message (RFC-822, STD-11, et cetera) objects
        - objects to parse and access MAIL message (environment and headers)
        - objects to parse UNIX mailbox
        - sub-objects (mail-message header)
        - manage writing of mail-message components
                a) email addresses
                b) general header folding

+ UNIX file-I/O library (essentially a replacement for UNIX STDIO;
        like "SFIO" from AT&T to name another)

+ other UNIX file I/O manipulation objects
        - for stream files (FILEBUF)
        - for mapped files (FILEMAP)

+ various interface software for access to standard UNIX databases
        - password-account
        - user-attribute (Oracle)
        - group
        - project (Oracle)
        - host (networking); both forward and reverse
        - network protocols
        - network services
        - UTMP (object oriented and non-object)
        - real-name indexed password DB (fast constant database
        object-oriented design)

+ other database access software (object oriented), supporting clusters
        - machine node-name
        - machine cluster-name

+ text indexing software (object oriented)
        - scans UNIX file-system for text-indexable files and indexes them to
          hash-table constant databases
        - constant database oriented
        - key extraction
        - index generation
        - query processing

+ logging (to files; object oriented)
        - logfile
        - logsys

+ ELF object-file management object (supplements standard UNIX object-file
                access routines)
        - OBJFILE object

== some general UNIX software and tools
Note: PCS - Personal Communications Services

+ text indexing and searching
        - key extraction (MKKEY)
        - index generation (MKINV)
        - query processing (MKQUERY)
        - output result display (MKTAGPRINT)
        - analysis (MKANALYSIS)

+ mail clients (part of PCS software facility)
        - one similar to MAILX
        - one a visual screen-oriented program
        - integrated name-address and mailing-list directory database

+ mail transport (part of PCS software facility)
        - network delivery transport switch (based on domain names)

+ other mail transport
        - RMAILER - transport agent client (proprietary protocol)
        - RMAILERD - transport server (proprietary protocol)

+ UNIX mail utilities
        - local delivery to spool (DMAIL)
        - local user mailbox delivery (DMAILBOX)
        - mail message injection (IMAIL); with address expansion
        - mailbox osrting and other processing (MBPROC)
        - mailbox mail message expiration (MAILEXPIRE, MBEXPIRE)
        - user mail alias access (MXALIAS)
        - system mail alias management (MAILALIAS)
        - create mail message (MKMSG)
        - extract header values and addresses (EMA)

+ bulletin-board (part of PCS software facility)
	- posting component
	- user configuration component
	- reading w/ mail
	- delivery and network transport software
	- maintenance and expiration

+ directory address information (part of PCS software facility)
	- lookup information on an email address
	- lookup information on a real-name

+ name-directory-service code (part of PCS software facility)
	- interface to various name-address directories
		› AT&T POST (names and mailing lists)
		› native (PCS) database (names and mailing lists)
		› UNIX password-account database
		› Sendmail 'alias' database
	- mail filtering by address

+ remote UNIX execution software ('RSHE')
	- used underlying RSH
	- passes environment and PWD to remote

+ remote UNIX execution software ('REX')
	- dynamically uses either underlying RSH or TCP/IP REXEC service
	- passes environment and PWD to remote

+ distributed UNIX execution software ('CEX')
	- designed and wrote general-purpose library code
	- uses a distributed dynamic database of machine load-averages
	- passes environment and PWD to remote

+ UNIX machine status maintenance libraries programs
	- a set of objects for accessing and maintaining UNIX machine status in a dynamic database
	- UNIX programs for accessing and updating local status to the database ('MSU', 'MSINFO')

+ builtin (loadable) commands for the Korn Shell (KSH)
	- these are eseentially builtin programs to the Korn Shell
	- some of these resemble standard UNIX utilities
	- these have stand-alone UNIX versions also

+ incremental backup utilities for UNIX
	- backs up files within directory trees that changed since the last backup; stored in a compressed format
	- create full backups
	- create incremental backups (generally hourly)
	- delete old backups

+ file linking and synchronization (and other file operations)
	- filefind
	- filelinker
	- filesyncer
	- filerm
	- filesize

= text document processing ('troff' family)
	› image inclusion
	› citation preprocessing ('referm' and 'mmcite')
	› cleaning (from MS and other bad sources), (TEXTCLEAN)
	› line inversion (LINEINVERT)
	› enhanced line folding (LINEFOLD)
	› typesetting text (TEXTSET)
	› typesetting cookies (COOKIESET)
	› citation-references (MMCITE, REFERM)

        › ROFF "tag" processing (GTAG)

= various network servers (TCPMUX, Telnet-like, login-plumbing)
        - TCPMUXD and FINGER server (proof of server architecture)

= enhanced UUCP family of software
        • including login daemon (PCSUUCPD)

= enhanced remote machine access:
        • rlogin
        • rsh
        • rcp

= web page generation software (specific application purposes)
        + MKARTICLES
        + HOMEPAGE
        + WEBCOUNTER
        + QUERYSTRING

= print spool submission (automatically formatted or semi-raw)
        - prt
        - prtfmt
        - prtdb (back-end database)
        - print device drivers (for print spoolers)

+ make tools
        - makesafe
        - makenewer
        - makedate
        - makeinstall
        - makebelow
        - makexxx

+ UNIX system management
        - sysdb
        - sysfs

+ UNIX account management
        - userinfo
        - groupinfo
        - projectinfo
        - pcsname
        - pcsorg
        - pcsprojinfo

= numeric conversion utilities
        - number conversion (common bases, roman numerals, words)
        - temperature comversion

+ specialized numeric calculator
        - factorial (nF)
        - exponential (nEk)
        - permutations (nPk) , with and without repetitions
        - combinations (nCk) , with and without repetitions

= database
        - program assembler instructions (database of)
        - Hammock branch detection (for program execution evaluation)
        - instruction (code) branch information (research)
        = subroutine call coverage (research)
        - inverted password (general UNIX account related)
        - name-address directory
        - document bibliographies
        - machine cluster status (for distributed processing)
        - mail message time-zone
        - mail message-id history

- name-server (PCS name-server daemon)
- calendar data
- text indexes (for searching)
- others