The Software Radio Architecture

As communications technology continues its rapid transition from analog to digital, more functions of contemporary radio systems are implemented in software, leading toward the software radio. What distinguishes software radio architectures? What new capabilities are more economically accessible in software radios than digital radios? What are the pitfalls? And the prognosis?

Joe Mitola

We are poised on the threshold of another revolution in radio systems engineering. Throughout the '70s and '80s radio systems migrated from analog to digital in almost every respect from system control to source and channel coding to hardware technology. And now the software radio revolution extends these horizons by liberating radio-based services from chronic dependency on hard-wired characteristics, including frequency band, channel bandwidth, and channel coding. This liberation is accomplished through a combination of techniques that includes multi-band antennas and RF conversion; wideband Analog to Digital (A/D) and Digital to Analog (D/A) conversion (A/D/A conversion); and the implementation of IF, baseband, and bitstream processing functions in general-purpose programmable processors. The resulting software-defined radio (or "software radio") in part extends the evolution of programmable hardware, increasing flexibility via increased programmability. And, in part, it represents an ideal that may never be fully implemented but that nevertheless simplifies and illuminates tradeoffs in radio architectures that seek to balance standards compatibility, technology insertion, and the compelling economics of today's highly competitive marketplaces.

Radio Architecture Evolution

Of the many possible definitions of architecture, the one that best relates systems, technology and economics is best suited to this discussion. We shall therefore define "architecture" as the comprehensive, consistent set of functions, components and design rules according to which systems of interest may be organized, designed, and constructed. A specific architecture entails a partitioning of functions and components such that functions are assigned to components and interfaces among components correspond to interfaces among functions.

When such functions and interfaces are defined in formal design rules via a public forum, the resulting architectures are called "open." The full economic benefits of open architectures require the existence of a large commercial base which sometimes fails to emerge in spite of openness. As system complexity increases, architecture becomes more critical because of its power to either simplify and facilitate system development (a "powerful" architecture) or to complicate development and impede progress (a "weak" architecture).

Radio architectures may be plotted in the phase space of network organization versus channel data rate, as shown in Fig. 1. These architectures have evolved from early point-to-point and relatively chaotic peer networks (e.g., citizens band and push-to-talk mobile military radio networks) toward more hierarchical structures with improved service quality. In addition, channel data rates continue to increase through multiplexing and spectrum spreading. In a multiple-hierarchy application, a single radio unit, typically a mobile terminal, participates in more than one network hierarchy. A software radio terminal, for example, could operate in a GSM network, an AMPS network, and a future satellite mobile network. Multiband multimode military radios and future Personal Communications Systems (PCS) that seamlessly integrate multimedia services across such diverse access modes represent the high end of that evolution and the focus of this discussion. The complexity of functions, components and design rules of these architectures continues to increase with each generation, as shown in Table 1. In particular, future seamless multimode networks will require radio terminals and base stations with agile RF bands, channel access modes, data rates, Bit Error Rates (BERs), power, and functionality. Software radios have emerged to increase quality of service through such agility. At the same time, software radio architectures simplify hardware component tradeoffs and provide new ways of managing the complexity of rapidly emerging standards.

The Canonical Software Radio Architecture

The components of the canonical software radio consist of a power supply, an antenna, a multiband RF converter, and a single chip containing

JOE MITOLA is a consulting scientist with The MITRE Corporation.

SPW is a registered trademark of the Alta Group.

TMS320 is a registered trademark of Texas Instruments, Inc.

Matlab is a registered trademark of The MathWorks Inc.

SPOX is a registered trademark of Spectrum Microsystem.

COSSAP is a registered trademark of SYNOPSIS, Inc.

Matcad is a registered trademark of MathSoft, Inc.

0163-6804/95/$04.00 1995 © IEEE

Generation	Illustrative Functions	Typical Components	Key Design Rules
Analog	Transmit and receive, channel select, squelch	Power, antenna, packaging (discrete analog baseband)	Channel allocations, power limits, standard modulations (AM, FM)
Early digital microwave	Transmit or receive protected modes, BER control	Analog + quadrature modems + Forward Error Control (FEC)	Analog + operations/management and bitstream multiplex interfaces, adjacent channel power envelopes
Analog Mobile Cellular Radio (MCR)	Analog + signaling and control	Analog + digital modems + embedded control processors	Analog + early digital + cell site and frequency plan + handoff protocol
Spread spectrum (CDMA/FH)	Code synchronization, code management, BER control	Analog + early digital + de-/spreading devices + embedded control processors	Analog + code design + peer network protocols + digital voice channel
TDMA MCR	Analog + early digital +analog MCR + spread spectrum + diversity + directivity	Analog + analog MCR + spread spectrum + multibeam antennas	Analog + early digital + analog MCR + digital voice channel + privacy and authentication
Future seamless multimode multimedia networks	Digital MCR + agile directional power management + data rate management + BER agility + mode handover + location reporting	Digital MCR + high programmability + agile modulators + multiband antennas + multiband RF	Digital MCR + mode handover criteria and protocols + end-to-end encryption + software defined services

Digital MCR = CDMA, TDMA, Frequency Hop (FH), and related hybrids.

Table 1. *Key elements of radio architecture evolution. Complexity of functions, components, and design rules increases with successive generations.*

A/D/A converters with an on-chip general purpose processor and memory that perform the radio functions and required interfaces illustrated in Fig. 2. The canonical mobile software radio terminal interfaces directly to the user (e.g., via voice, data, fax, and/or multimedia). The canonical base station interfaces to the public switched telephone network (PSTN). Fully instrumented base stations support operations and maintenance, developers and researchers via services development workstation(s). The placement of the A/D/A converters as close to the antenna as possible and the definition of radio functions in software are the hallmarks of the software radio. Thus, although software radios use digital techniques, software-controlled digital radios are generally not software radios. The key difference is the total programmability of software radios, including programmable RF bands, channel access modes, and channel modulation.

Contemporary radio designs mix analog hard-

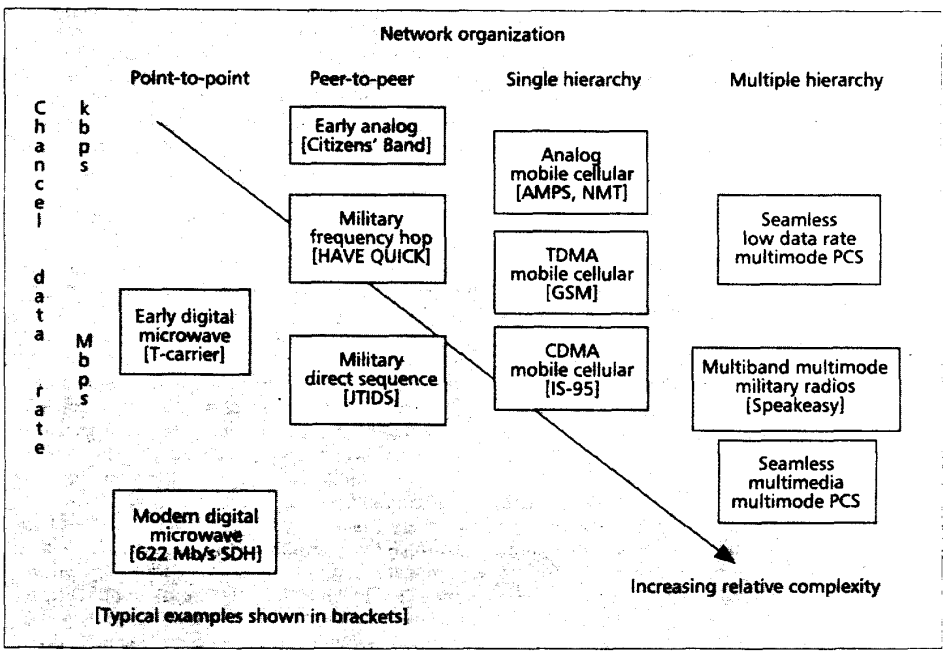
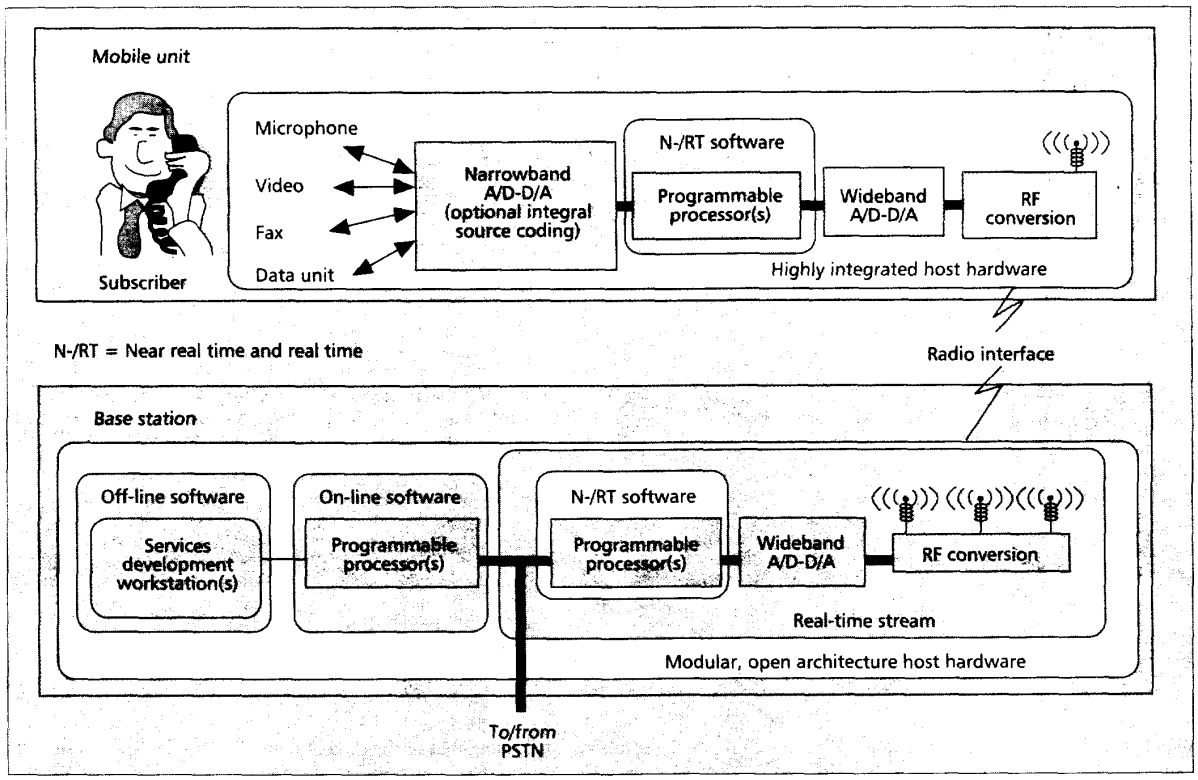


Figure 1. *Radio architectures are evolving towards high levels of complexity.*

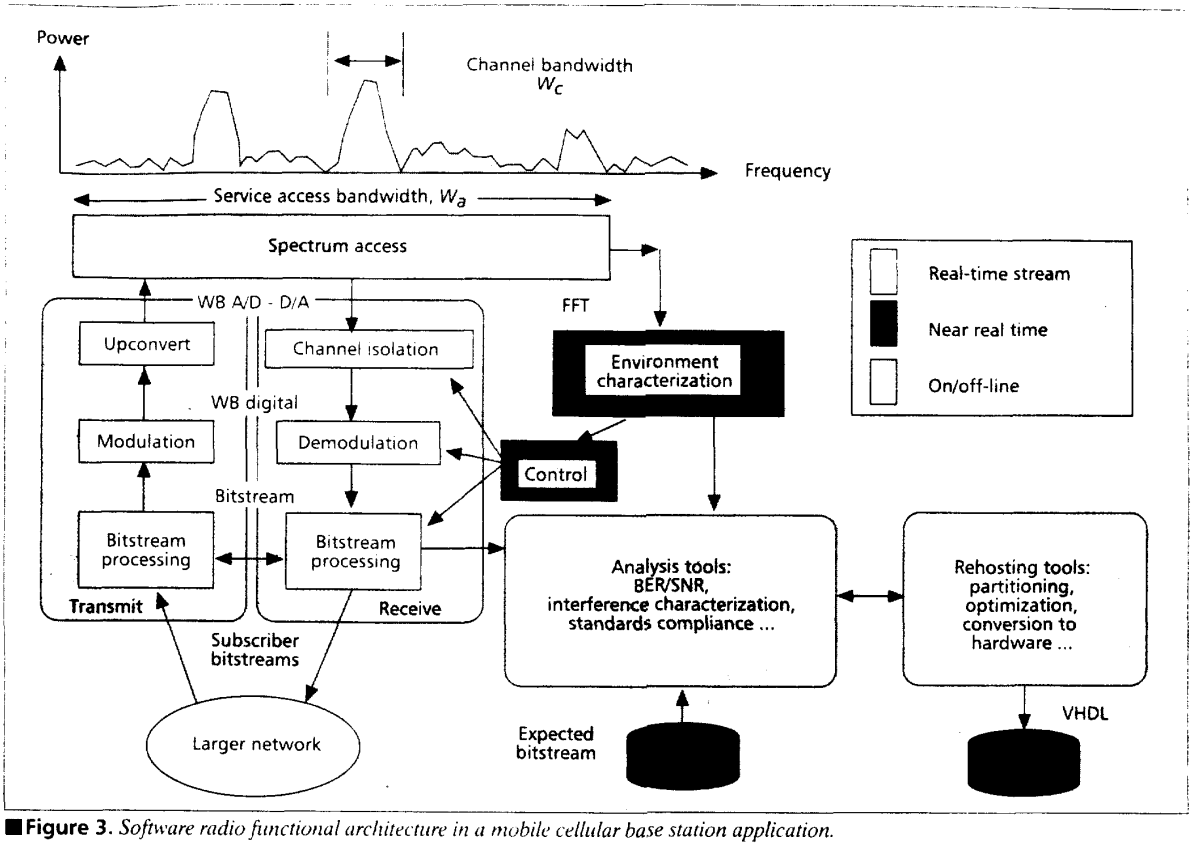


■ **Figure 2.** *In the canonical software radio, hardware is simple and functions are software-defined.*

ware, digital hardware, and software technologies. It is instructive to consider software radios *per se* to better understand benefits, pitfalls and relationships to other technologies. Software radios have become practical as costs per millions of instructions per second (MIPS) of digital signal processors (DSPs) and general purpose central processor units (CPUs) have dropped below U.S. $10 per MIPS. The economics of software radios become increasingly compelling as demands for flexibility increase while these costs continue to drop by a factor of two every few years. At the same time, absolute capacities continue to climb into the hundreds of millions of floating-point operations per second (MFLOPS) per chip, at which point software radios are compatible with commercial TDMA and CDMA applications. In addition, A/D/A converters available in affordable single-board open architecture configurations offer bandwidths of tens of MHz with the dynamic range required for software radio applications [1]. Multimedia requirements for the desktop and palmtop continue to exert downward pressure on parts count and on power consumption of such chip sets, pushing the software radio technology from the base station to the mobile terminal. Although the tradeoffs among analog devices, low-power ASICs, DSP cores and embedded microprocessors in handsets remain fluid, cutting-edge base stations employ software radio architectures. Finally, the multiband, multimode flexibility of software radios appears central to the goal of seamless integration of PCS, land mobile and satellite mobile services (including truly nomadic computing) toward which many of us aspire.

But software radio engineering is fraught with pitfalls. It is difficult to engineer wideband, low-loss antennas and RF converters. It is also difficult to accurately estimate processing demand of applications and processing capacity of reprogrammable DSP/CPU configurations. In addition, sustaining required data rates across interprocessor interfaces is problematic. There are several vendor-unique high capacity wideband signal buses, but open architecture standards for this critical element of the software radio architecture have not yet emerged. DSP function libraries continue to expand, and block diagram-based "integrated environments" exist. But we do not yet have the ability to mix and match real-time software tools and modules from different software suppliers as we can mix and match VME boards today. Most of the pitfalls can be avoided, however. And continued progress in these difficult areas will further reduce costs and time to market.

This article therefore provides a tutorial review of software radio architectures and technology, highlighting benefits, pitfalls, and lessons learned. This includes a closer look at the canonical functional partitioning of channel coding into antenna, RF, IF, baseband, and bitstream segments. A more detailed look at the estimation of demand for critical resources is key. This leads to a discussion of affordable hardware configurations, the mapping of functions to component hardware, and related software tools. This article then concludes with a brief treatment of the economics and likely future directions of software radio technology. Companion articles in this issue address the critical technologies of A/D convert-



■ **Figure 3.** *Software radio functional architecture in a mobile cellular base station application.*

ers and DSP cores and cutting-edge applications in greater detail.

Software Radio Overview

The software radio architecture is widely applicable to trunk radios, peer networks, air and sea traffic management, mobile military communications, and satellite mobile systems. For simplicity, this overview describes the software radio architecture in a mobile cellular/PCS setting.

In an advanced application, a software radio does not just transmit: it characterizes the available transmission channels, probes the propagation path, constructs an appropriate channel modulation, electronically steers its transmit beam in the right direction, selects the appropriate power level, and then transmits. Again, in an advanced application, a software radio does not just receive: it characterizes the energy distribution in the channel and in adjacent channels, recognizes the mode of the incoming transmission, adaptively nulls interferers, estimates the dynamic properties of desired-signal multipath, coherently combines desired-signal multipath, adaptively equalizes this ensemble, trellis decodes the channel modulation, and then corrects residual errors via forward error control (FEC) decoding to receive the signal with lowest possible BER.

Finally, the software radio supports incremental service enhancements through a wide range of software tools. These tools assist in analyzing the radio environment, defining the required enhancements, prototyping incremental enhancements via software, testing the enhancements in the radio environment, and finally delivering the service enhancements via software and/or hardware.

The Real-Time Channel Processing Stream

The canonical software radio architecture includes the channel processing stream, the environment management stream and associated software tools illustrated in Fig. 3. The real-time channel processing stream incorporates channel coding and radio access protocols. Channel processing is characterized by discrete time point-operations such as the translation of a baseband signal to an intermediate frequency (IF) by multiplying a discrete time-domain baseband waveform by a discrete reference carrier to yield a sampled IF signal. The time between samples is on the order of tens of microseconds to hundreds of nanoseconds. Such point-operations require hundreds of MIPS and/or MFLOPS to Giga-FLOPS with strictly isochronous performance. That is, sampled data values must be computationally produced and consumed within timing windows on the order of the time between samples in order to maintain the integrity of the signals represented therein. Input/output (I/O) data rates of this stream approach a gigabit per second per A/D converter. Although these data rates are decimated through processing, it is challenging to sustain isochronism through I/O interfaces and hard real-time embedded software in this stream. Multiprocessing is

Technology limitations that require hardware-based delivery are overcome by mapping critical elements of the service enhancement to hardware via VHDL.

therefore best organized as a pipeline with sequential functions of the stream assigned to serially interconnected processors, i.e., a multiple instruction multiple data-stream (MIMD) multiprocessing architecture [2].

The Environment Management Stream

The near-real-time environment management stream continuously characterizes radio environment usage in frequency, time and space. This characterization includes channel identification and the estimation of other parameters such as channel interference levels (depending on the specific signaling and multiple access scheme) and subscriber locations. The environment management stream employs block operations such as fast Fourier transforms (FFTs), wavelet transforms, and matrix multiplies for beam forming. Channel identification results are needed in times on the order of hundreds of microseconds to hundreds of milliseconds, while power levels may be updated in milliseconds and subscriber locations may be updated less frequently. The block structure of such operations is readily accommodated by a MIMD parallel processor. The interface between this highly parallel environment management stream and the pipelined channel processing streams must synchronize the environment management parameters to the channel processing streams.

On-Line and Off-Line Software Tools

On-line and off-line systems analysis, signal processing, and rehosting tools illustrated in Fig. 3 allow one to define incremental service enhancements. For example, an enhanced beamformer, equalizer and trellis decoder may be needed to increase subscriber density. These enhancements may be prototyped and linked into the channel processing stream, allowing one to debug the algorithm(s), to experiment with parameter settings, and to determine the service value (e.g., in improved subscriber density) and resources impact (e.g., on processing resources, I/O bandwidth, and time delays).

Software-based enhancements may be organized around managed objects, collections of data and associated executable procedures that work with object resource brokers and conform to related open architecture software interface standards such as the Common Object Resource Broker (CORBA) [3]. Enhancements may then be delivered over the air to other software radio nodes, as contemplated in the future software-defined telecommunications architectures being considered by ITU-T and embraced by NTT [4] and others [5, 6]. A well integrated set of analysis and rehosting tools leads to the creation of incremental software enhancements relatively quickly, with service upgrades provided over-the-air as software-defined networks proliferate. Technology limitations that require hardware-based delivery are overcome by mapping critical elements of the service enhancement to hardware via VHDL.

Partitioning of The Channel Processing Stream

The classical canonical model of communications concatenates source encoder, channel encoder, channel (which adds noise, interference, and distortion), channel decoder, and source decoder. The channel encoder/decoder and related radio access functions constitute the real-time stream. The canonical software radio architecture partitions classical channel coding and decoding into the channel access segments of Fig. 4. These segments are: antennas, RF conversion, IF processing, baseband processing and bitstream processing. This canonical partitioning is useful because of the significant differences in functionality between segments, because of the strong cohesion among functions within a segment; because of large changes in bandwidth due to decimation within a segment; and because of the ease with which these particular segments are mapped to affordable open-architecture hardware. Further rationale for this partitioning is based on set-theoretic considerations [7]. This partitioning also structures the estimation of first-order resource requirements so that they may be combined in ways that accurately predict system performance.

The Antenna Segment

The antenna(s) of the software radio span multiple bands, up to multiple octaves per band with uniform shape and low losses to provide access to available service bands. In military applications, for example, a mobile terminal may need to employ VHF/UHF line of sight frequencies, UHF satellite communications, and HF as a backup mode. Switched access to such multiple bands requires octave bandwidth antennas and/or multiple antennas per band and an agile frequency reference in the RF Segment. In addition, multiple antenna elements may be part of a beam forming network for interference reduction or space division multiple access (SDMA).

The relationship between interference cancellation capacity and the number of antenna elements varies. A single auxiliary element, for example, can reduce interference of a large number of interferers (e.g., those in the "back lobe"). Algorithms that reduce interference through non-spatial techniques (e.g., cyclostationary algorithms) can also reduce a large number of interferers with one or with no auxiliary antenna elements. Beam forming of N antenna elements can place $N-1$ adaptive nulls on interferers sufficiently separated in azimuth, but coherent multipath may require a distinct null for each distinct multipath direction, reducing the number of interferers accordingly. Polarization scrambling from nearby reflecting surfaces may require $2N+1$ elements for N paths. The structure of the antenna array(s) determines the number of distinct physical and logical signal processing paths in the RF conversion and IF processing segments. As a result, the competing demands for directional selectivity, multipath compensation and interference suppression versus wideband low-loss antennas versus affordability define the tradeoffs of the antenna segment.

The RF Conversion Segment

RF conversion includes output power generation, preamplification, and conversion of RF signals to and from standard intermediate frequencies (IFs) suitable for wideband A/D/A conversion. In most radio bands, RF conversion will be analog. Certain critical RF problems are exacerbated in the software radio. These include the need for amplifier

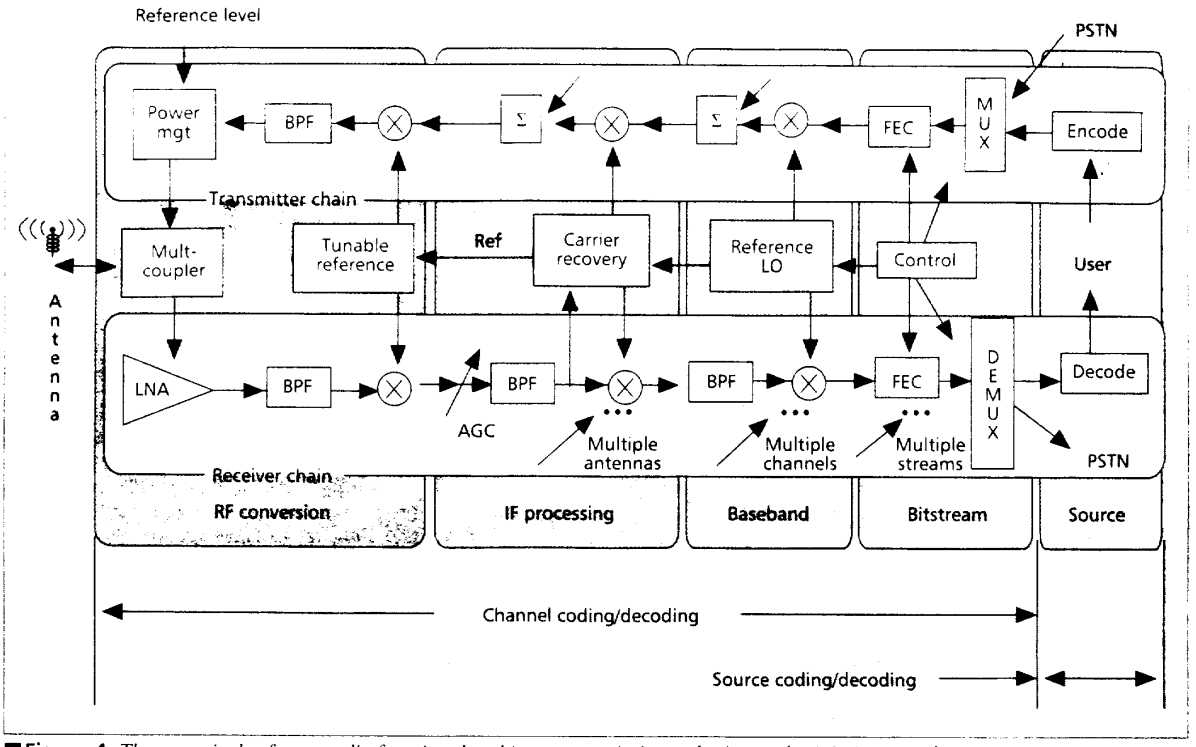


Figure 4. *The canonical software radio functional architecture maximizes cohesion and minimizes coupling.*

linearity and efficiency across the access band. RF shielding of processors may also be necessary to avoid the introduction of processor clock harmonics into the analog RF/IF circuits. Cositing of multiple transmitters also creates electromagnetic interference (EMI) problems, but these are about the same for software radios as for cosite collections of multiple discrete hardware radios.

Placement of the A/D/A Converters Is Key

W_a, the bandwidth of the IF to be digitized, determines what kinds of A/D techniques are feasible. According to the Nyquist criterion for band limited signals f_s, the sampling rate of the A/D converter, must be at least twice W_a. Practical systems typically require modest oversampling:

$$f_s > 2.5\, W_a$$

Wideband A/D/A converters access broad instantaneous segments of spectrum, typically 10 to 50 MHz. Such wide access may also be achieved in parallel subbands of more modest 1 to 10 MHz bandwidths each. The dynamic range of each parallel subband depends on the dynamic range of the A/D/A converters. Since the product of dynamic range times sampling rate is approximately constant for a given A/D/A technology, narrower subbands generally increase the useful dynamic range, albeit at the cost of increased system complexity. The many issues related to filtering and A/D conversion are deferred to the companion article on A/D converters in this issue [1].

The placement of wideband A/D/A conversion before the final IF and channel isolation filters achieves three key architectural objectives:

- It enables digital signal processing before detection and demodulation.
- It reduces the cost of mixed channel access modes by consolidating IF and baseband processing into programmable hardware.
- It focuses the component tradeoffs to a single central issue: providing the computational resources (I/O bandwidth, memory, and processing capacity) critical to each architecture segment, subject to the size, weight, power, and cost constraints of the application.

The IF Processing Segment

The IF processing segment maps the transmit and receive signals between modulated baseband and IF. The IF receiver processing segment includes wideband digital filtering to select a service band from among those available. Furthermore, IF filtering recovers medium band channels (e.g., a 200 kHz TDMA channel in GSM) and/or wideband subscriber channels (e.g., a 2 MHz CDMA channel) and converts the signal to baseband. The complexity of frequency conversion and filtering is the first order determinant of the processing demand of the IF segment. In a typical application, a 12.5 MHz mobile cellular band is sampled at 30.72 MHz (M samples per second). Frequency translation, filtering and decimation requiring 100 operations per sample equates to more than 3000 MIPS of processing demand. Although such microprocessors are on the horizon, contemporary implementations offload this computationally intensive demand to dedicated chips such as the Harris Decimating Down Converter (DDC) or Gray digital receiver chip. Spreading and de-spreading of CDMA, also an IF processing function, creates demand that is proportional to the bandwidth of

Application	Radio function	Segment	First order demand drivers
Analog Voice Channel (Receiver)	Companding	Source	Speech bandwidth (W_v) and sampling rate
	Gap suppression	Bitstream	Gap identification algorithm
	FM modulation	Baseband	Interpolation required (W_{fm}/W_v)
	Up conversion	IF	IF carrier, f_i, and FM bandwidth: $W_i = W_{fm}$
	Band selection	IF	Access bandwidth (W_a)
	Channel selection	IF	Channel bandwidth (W_c)
	FM demodulation	Baseband	f_i, W_{fm}
	DS0 reconstruction	Bitstream	Speech bandwidth
TDMA (TDM)	Voice codec	Source	Choice of voice codec
	FEC coding	Bitstream	Code rate, algorithm complexity
	Framing	Bitstream	Frame rate (R_f)
	MSK modulation	Baseband	Baud rate (R_b)
	Up conversion	IF	f_i, $R_b/2$
(Receiver)	Band selection	IF	Access bandwidth (W_a)
	Channel selection	IF	Channel bandwidth (W_c)
	Demodulation	Baseband	Baud rate (R_b) or channel bandwidth (W_c)
	Demultiplexing	Bitstream	Frame rate (R_f)
	FEC decoding	Bitstream	Code rate, algorithm complexity
	DS0 reconstruction	Source	Voice codec

Table 2. *Illustrative functions, segments, and resource demand drivers.*

the spreading waveform (typically the chip rate) times the baseband signal bandwidth. This is so computationally intensive that with current technology limitations, it is typically assigned to dedicated chips as well.

The Baseband Processing Segment

The baseband segment imparts the first level of channel modulation onto the signal (and conversely demodulates the signal in the receiver). Predistortion for nonlinear channels would be included in baseband processing. Trellis coding and soft decision parameter estimation also occur in the baseband processing segment. The complexity of this segment therefore depends on the bandwidth at baseband W_b, the complexity of the channel waveform, and the complexity of related processing (e.g., soft decision support). For typical digitally encoded baseband waveforms such as binary phase shift keying (BPSK), quadrature phase shift keying (QPSK), Gaussian minimal shift keying (GMSK), and 8-PSK with channel symbol (baud) rates of R_b:

$$R_b/3 < W_b < 2^*R_b$$

In the transmission side of the baseband segment, such waveforms are generated one sample at a time (a "point operation"). If three samples are generated for the highest frequency component, demand falls between R_b and 6^*R_b. Greater oversampling decreases the transmitted power of spectral artifacts, but also increases transmit power and processing demand. In the receiver, digital baseband modulations require timing recovery which typically includes the integration of baud intervals over time. If baud interval is measured in transitions of a high-speed clock, some timing-sensitive signal structures (e.g., TDMA) and some synchronization algorithms require up to 96 b precision integer arithmetic in the clock recovery loop(s), and such extended precision arithmetic may not be readily available, particularly on newer chips. Analog baseband modulation, such as FM voice, may also be encoded and demodulated in software in the baseband segment, with a processing demand of less than 1 MIPS per subscriber. Such software simulation of analog modulations helps achieve backwards compatibility with analog standards.

The Bitstream Segment

The bitstream segment digitally multiplexes source-coded bitstreams from multiple users (and, conversely, frames and demultiplexes them). The bitstream segment imparts forward error control (FEC) onto the bitstream, including bit interleaving and block and/or convolutional coding and/or automatic repeat request (ARQ) detection and response. Frame alignment, bit-stuffing, and radio link encryption occur in the bitstream segment. Encryption requires the isolation of encrypted bits from clear bits, resulting in the requirement to partition and isolate bitstream hardware accordingly. Final trellis-coded modulation (TCM) decisions occur in the bitstream segment. Final TCM converts soft/delayed decision parameters from the baseband segment to final bit decisions. The complexity of this segment depends on multiplexing, framing, FEC, encryption, and related bit manipulation operations.

Signaling, control and operations, administration and maintenance (OA&M) functions are also provided in the bitstream segment. The demand associated with these functions depends on the signaling, control and operations systems. Demand increases linearly with the number of simultaneously active subscribers. These functions are event-driven and typically impart an order of magnitude less computational demand than baseband processing. These functions, may, however, require access to distributed data bases, not all of which will be local to the base station. Thus, although the processing demand is relatively small, the timing requirements may be severe.

The Source Segment

The source segment differs between the mobile terminal and the base station. In the mobile terminal, the source segment consists of the user and the source encoders and decoders. Here, the relatively narrowband voice and fax A/D/A converters are typically located in the handset, palmtop, or workstation. In the base station, on the other hand, the source segment consists of the interface to the PSTN for access to remote source coding. Conversion of protocols required for interoperability with the PSTN creates processing demand in the base station's source segment. Conversion of DS0 64 kb/s PCM to RPE-LTP (GSM [11]), for example, would create 1 to 2 MIPS of demand per subscriber.

End-to-End Timing Budgets

Time delays are introduced in the IF, baseband, bitstream and source segments due to finite I/O and processing resources that empty and fill buffers in finite but sometimes random amounts of time. The end-to-end accumulation of these delays must be kept within bounds of isochronism at each segment-to-segment interface. These bounds depend on signal type and the larger network architecture. Thus, for example, end-to-end voice delay should be less than 150 ms, but the external network may consume 100 ms of this timing budget, leaving only 50 ms for the software radio. Maintaining such budgets in software radios is complicated (compared to digital radios) by queuing delays internal to and between processors.

Typical application	RF *fc*	*Wa*	IF *Wi*	Channel code	Baseband *Wi*	Bitstream States	Bitstream FEC	Multiplexing	Privacy	Source
FM mobile	VHF	30 kHz	30 kHz	FM	4 kHz	Inf	—	(PTT)	None	Compand
MCR voice	UHF	25 MHz	25 kHz	FM	4 kHz	Inf	—	(CA)	None	Compand
MCR control		"	"	MSK	10 kb/s	4	Conv	TDM	None	Data
GSM	UHF	25 MHz	200 kHz	GMSK	270.83 kb/s 13 kb/s	4	CPF	TDM	Encrypt	RPE-LTP
Mil FH	VHF	60 MHz	30 kHz Agile	FH-QPSK	16 kb/s	4	Conv ARQ	(PTT)	Encrypt	LPC/Delta
CDMA	UHF	175 MHZ	50 MHz	M-PSK	8 kb/s	M	Conv	(CD)	Privacy	LPC
JTIDS [12]	L-Band	250 MHz	3 MHz	MSK	150 kb/s	4	Multiple	TDM	Encrypt	LPC
Microwave	SHF	20 MHz	20 MHz	64 QAM	90 Mb/s	64	Multiple	TDM	Bit Stuff	PCM

(Implicit Multiplexing): (PTT) = Push to Talk; (CA) = Circuit Assigned; (CD) = Code Division
CPF: GSM specifies various modes of Convolutional, Parity and Fire codes [8].

■ **Table 3.** *Critical applications parameters bound offered demand.*

Estimating Resource Requirements

Resources critical to the software radio architecture include I/O bandwidth, memory and processing capacity. Good estimates of the demand for these resources results in a well informed mapping of the above segments to appropriate hardware. Depending on the details of the hardware, the critical resource may be memory, bus, or I/O bandwidth, or a particular embedded processor. Identification of the critical resource can be accomplished quickly using the techniques highlighted in this section. When such critical resources are identified early in the design process, order of magnitude shortfalls in performance can be avoided. By identifying first-order demand drivers; aggregating demand for each major system resource; comparing demand systematically to hardware/software capacity; and then managing the critical resources appropriately, such shortfalls can be avoided.

Standardized Measures of Demand and Capacity

Since many contemporary processors include pipelined floating point arithmetic sections or single instruction butterfly operations, MIPS and MFLOPS are *not* interchangeable. Both types of operations, however, require processor clock cycles, allowing one to express demand in a common measure of millions of operations per second (MOPS) where an operation is the amount of work that can be accomplished by a given resource (CPU, floating point unit, etc.) in a single clock cycle of a standard width (e.g., 32 b). Using this measure, required MIPS and MFLOPS may both be expressed in MOPS. In addition, I/O, direct memory access, auxiliary I/O processor throughput, memory and bus bandwidths may all be expressed in MOPS where the operand is a standard data word and the operation is store or fetch. MOPS may then be accumulated independently for each potentially critical resource (CPU, DSP unit, floating point processor, I/O bus, memory bank, etc.). Finally, software demand may be translated rigorously to equivalent MOPS. Benchmarking is the key to this last step since hand coded assembly language algorithms often out-perform high order language (HOL) code by an order of magnitude and HOL often outperforms code-generating software tools by an order of magnitude. Rigorous analysis of performance is accomplished by comparing demand and capacity in standards MOPS for each resource using queueing theory. This identifies bottlenecks and yields useful estimates of performance.

Estimate Demand In The Context of The Canonical Data Flow

Table 2 shows how functional architecture parameters drive the resource demand of the associated segment in illustrative applications. Although the associated demand may exceed the capacity available with a given generation of devices, the capacity estimates serve to identify the hardware that best supports a given segment. By determining the number of operations required per point-operation (e.g., a filtering stage) and multiplying by the critical parameter (e.g., the data rate of the stream being filtered), one can quickly arrive at demand estimates that frame the related implementation decisions.

Table 3 shows critical parameters of processing demand for illustrative applications. The demand depends to a first-order approximation on the signal bandwidths and on the complexity of key operations within IF, baseband, bitstream, and source segments as follows:

$$D = D_{if} + N^*(D_{bb} + D_{bs} + D_s) + D_o$$

Where D_{if}, the IF processing demand, is proportional to the acquisition bandwidth, and to the complexity of the service band isolation filter and of the subscriber channel isolation filter(s). N is the maximum number of simultaneous subscribers to be serviced by the node. The baseband demand D_{bb} is proportional to the bandwidth of a single subscriber channel and the complexity of demodulation. The bitstream demand D_{bs} is proportional to the data rate and the complexity of the forward error control algorithm, bit stuffing/interleaving, signaling and control and miscellaneous operations functions. The source segment demand D_s depends on the complexity of the

interface to the PSTN. And finally, the management overhead processing demand D_o depends on the signaling system, the radio access network overhead and on the degree of instrumentation that is provided to OA&M, developers and/or researchers.

Table 4 shows how key parameters and processing demand are related for an illustrative application. This demand must be met by processors with sufficient capacity to support real-time performance, including isochronism and end-to-end delay budgets. At present, most IF processing is off-loaded to special-purpose digital receiver chips because general purpose processors with sufficient MOPS are not yet cost effective compared to specialized hardware. This tradeoff changes approximately every 18 months, with the trend in favor of the general-purpose processor. In the table, the total baseband plus bitstream processing demand of 4.7 MOPS per user is within the capabilities of nearly all DSP chips. As a result, several subscribers may share a single DSP chip. Aggregate demand (all users, including overhead) of 142.6 MOPS is nominally within the capacity of a Quad TMS320 C50 board, but when multiplexing more than one user's stream into a single processor, memory buffer sizes, bus bandwidth and fan-in/fan-out may be more critical than processing capacity.

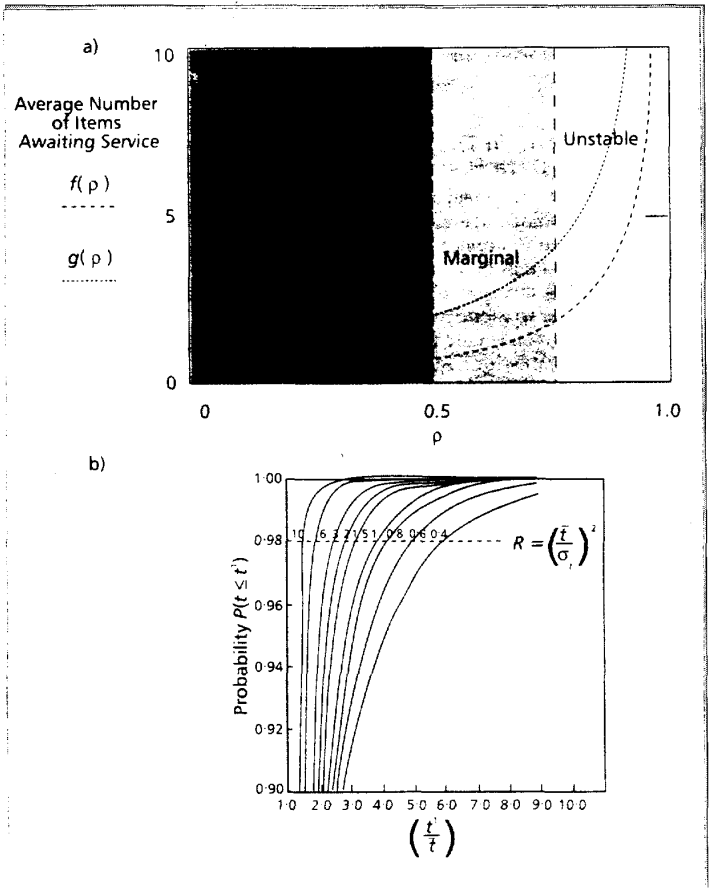


Figure 5. *a) Facility utilization determines performance, underscoring the criticality of demand estimates; b) ratio of variance to mean determines reliability, $P(t \leq t')$ for specific offered demand [Gamma distribution courtesy of Tebbs & Collins ©1977, McGraw-Hill (UK), reproduced with permissionof McGraw-Hill].*

Facility Utilization Accurately Predicts Performance

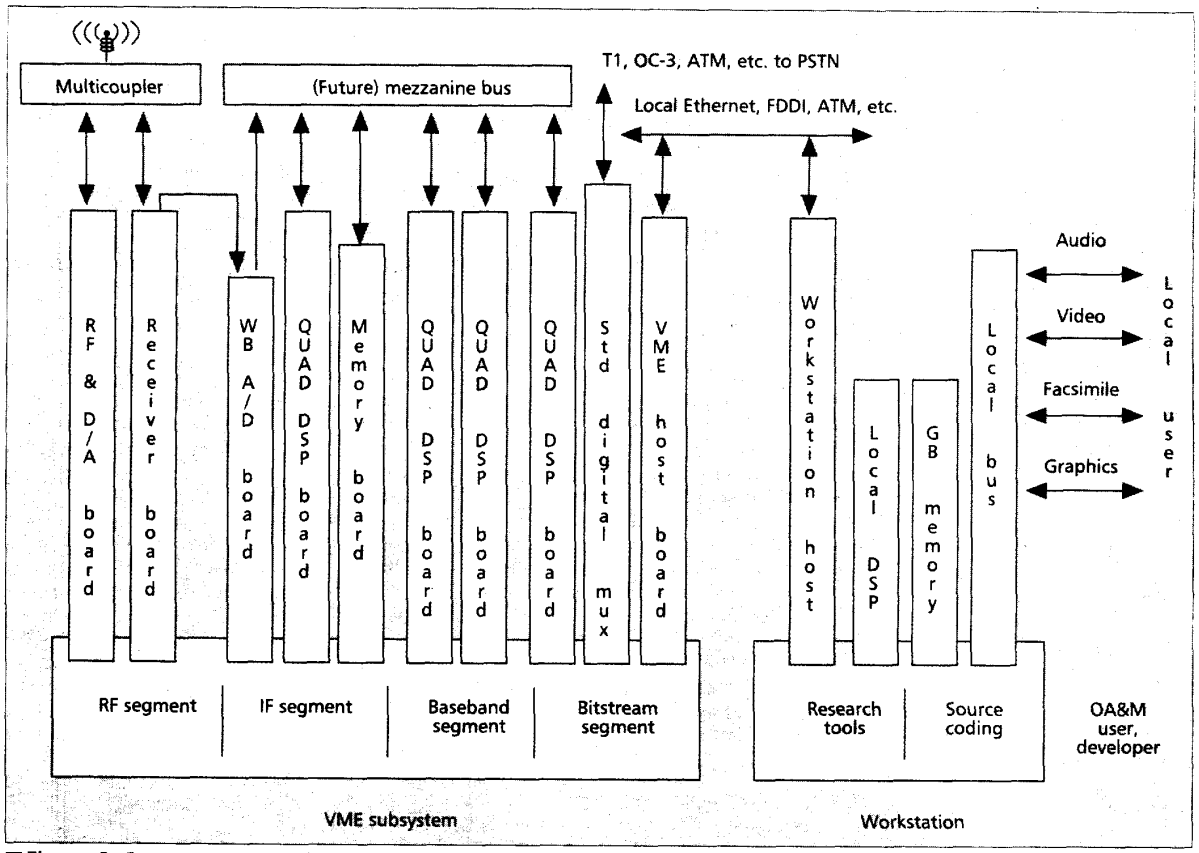
The most significant design parameter of the mapping of processing demand to processor capacity is resource utilization (of the CPU, DSP chip, memory, bus, etc.). Resource utilization, ρ, is the ratio of offered demand to available capacity. Figure 5a shows how the average number of items awaiting service at the resource varies as a function of utilization. (See [9] for an accessible treatment of material covered at a deeper level in [8] and [10].) The curve $f(\rho)$ represents exponentially distributed service times, while $g(\rho)$ represents uniform service times; system performance generally lies on or between these two curves. Rarely does usable processing capacity exceed 50 percent of stand-alone capacity as indicated by operating regions. Robust performance occurs when ρ is less than 0.5. The undesired events that result in service degradation will occur with noticeable regularity for $0.5 < \rho < 0.75$. For $\rho > 0.75$, the system is generally unstable, with queue overflows regularly destroying essential information.

An analysis of variance is required to establish the risk that the required time delays will be exceeded. The incomplete Gamma distribution of Fig. 5b characterizes the risk of exceeding a specified delay through the ratio of the specified delay to the average delay (under various relationships of the mean to the variance). Software radios work well if there is a 98 percent probability of staying within specified performance using the curve for $R = 1$ in the figure.

The simplified analysis presented above illustrates the key ideas with the processor as the critical resource. But in many cases, I/O, the backplane bus or memory access will be the critical resource. The approach is readily adapted to any critical resource. Simulation can be used to refine the estimates obtained from this simple model. Management of the system bottlenecks requires the further insights gained through benchmarking and rapid prototyping. But there is no free lunch. Software radios require up to ten times the raw processing power of ASICs and special purpose chips of the same technology. Software radios therefore lag special purpose hardware implementations by about one or two hardware generations. Thus, software radio architectures have appeared first in base stations using heterogeneous multiprocessing hardware, which is discussed briefly in the following section.

Heterogeneous Multiprocessing Hardware

Segment boundaries among antennas, RF, IF, baseband, bitstream, and source segments evolved because they make it easy to build software radios on parallel, pipelined, heterogeneous multiprocessing hardware. Such partitioning maps the segments to affordable open hardware architectures, as illustrated in Fig. 6. In this example, the VME chassis hosts the RF, IF, baseband, and bitstream segments, while the workstation houses the user interface, research tools, development



■ **Figure 6.** *Open component architecture supplies processing capacity with affordable technology insertion.*

tools and local source coding/decoding (Table 5).

In the most cost effective leveraging of open architectures, VME systems are configured by experienced developers as heterogeneous multiprocessors. The VME host serves as systems control processor . The DSP processors support the real-time channel processing stream, sometimes configured as one DSP per subscriber channel. The path from A/D to the first filtering/decimation stage typically uses a dedicated point-to-point mezzanine interconnect such as DT Connect™ (Data Translation). Timing the data transfers across this bus with the point operations of the first filtering and decimation stage introduces inefficiencies that reduce throughput. Fan-out from IF processing to multiple baseband processing DSPs is also typically accomplished via a dedicated point-to-point mezzanine path. Current implementations use customized fiber channel, TMS320/C50 or transputer links. As suggested in the figure, however, an open architecture optical mezzanine bus or local ATM switch should supplant today's vendor-unique point-to-point interconnects. An open architecture mezzanine bus with a capacity of 125 MHz x 16 bits per channel with an aggregate capacity of eight simultaneous channels would fill a critical void in contemporary open architecture standards.

Instead of configuring such heterogeneous multiprocessors at the board level, one might chose a preconfigured system. Mercury™, for example, offers a mix of SHARC 21060 (Analog Devices), PowerPC RISC, and Intel i860 chips with a unique Raceway interconnect with nominally three paths at 160 MB/sec interconnect capacity. Arrays of WE32s were used in AT&T's DSP-3 system. Arrays of i860s are available from Sky, CSPI, and others. Of particular note is the militarized TOUCHSTONE system which is also based on the i860. Such turnkey systems integrate block programming languages, automatic translation to C

Segment	Parameter	Illustrative value	Demand estimate
IF	W_a	10 MHz (2.5 oversampling)	
	IF Filter	100 OPS/ Hz	D_{if} = 2500 MOPS*
Users	N	30/ cell site	
Baseband	W_c	30 kHz	
	Demodulator	50 OPS/Hz	D_{bb}= 1.5 MOPS
Bitstream	R_b	32 kb/s	
	FEC, Signaling	100 OPS/b/s	D_{bs} = 3.2 MOPS
Source	CELP Codec	1.6 MOPS/user	D_s = 1.6 MOPS/user
Signaling	SS7	2 MOPS/site	D_o = 2 MOPS
Aggregate	DSP MOPS		D = 142.6 MOPS per cell site

* D_{if} is off-loaded to dedicated digital hardware and is therefore not included in D.

■ **Table 4.** *Illustrative processing demand: analog mobile cellular base station.*

Segment	Module	Characteristics	Illustrative manufacturers
RF	RF/IF	HF, VHF, UHF	Watkins Johnson, Steinbrecher
(A/D/A)	A/D	1 to 70 Msa/sec	Analog Devices, Pentek, DT
IF, BB	DSP	4 x 30 MFLOPS	AMD, TI, Intel, Mercury, Sky
IF	Digital Rx	30 72 MHz filters	Harris, Gray, E-Systems
IF	Memory	64 MB at 30 72 MHz	TI, Harris, TRW
BS, SC	Hosts	M680 x 0	Motorola, Force, Intel
SC	Workstation	50 to 100 Specmarks	Sun, Hewlett-Packard, DEC

(BB = Baseband; BS = Bitstream; SC = Source Coding)

■ **Table 5.** *Mapping of segments to open architecture VME modules.*

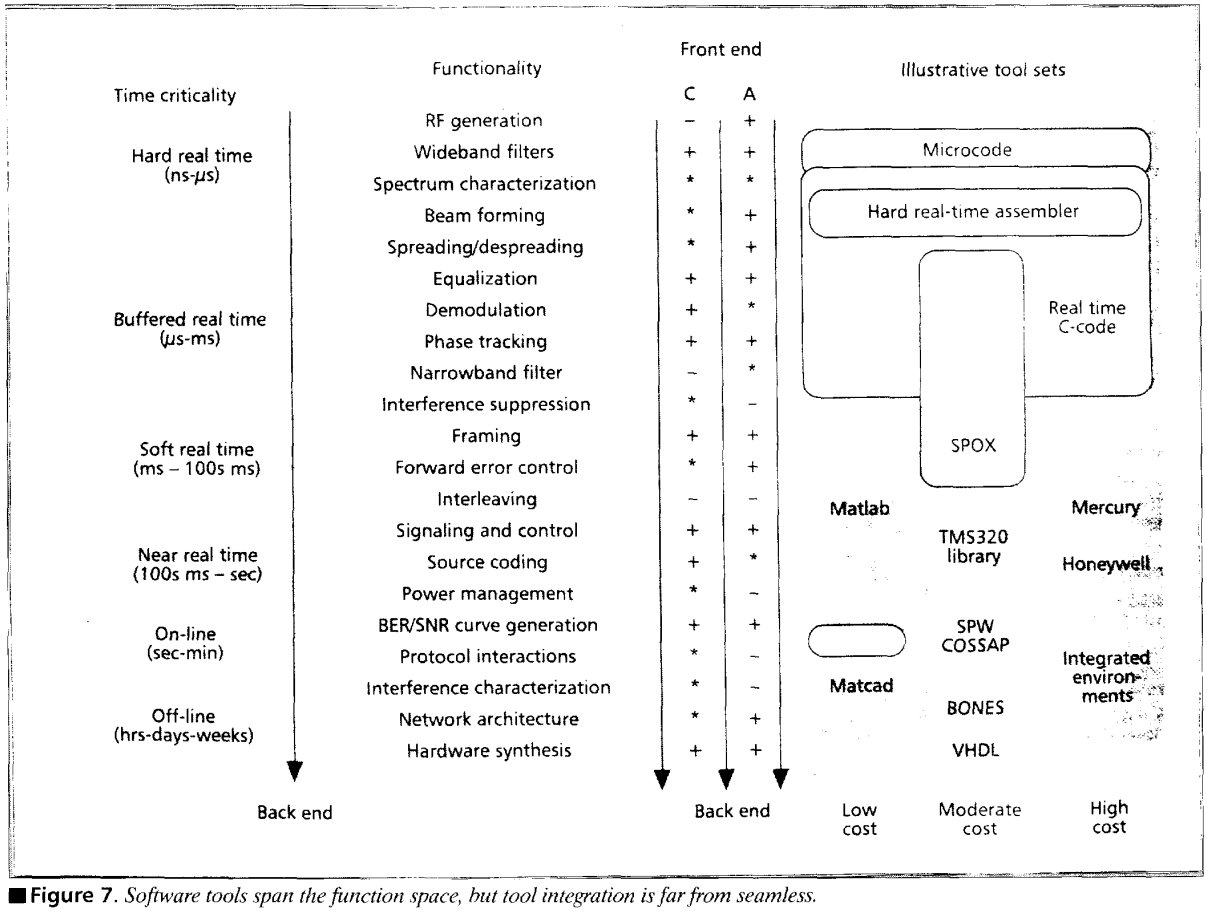
and machine code and extensive debugging support but at relatively high cost of entry. Vendor-unique turnkey systems may be excellent choices when there is a good match between the application and the vendor's standard configured system. Vendor-unique details tend to limit hardware and software choices to the original vendor.

Architecture Tradeoffs

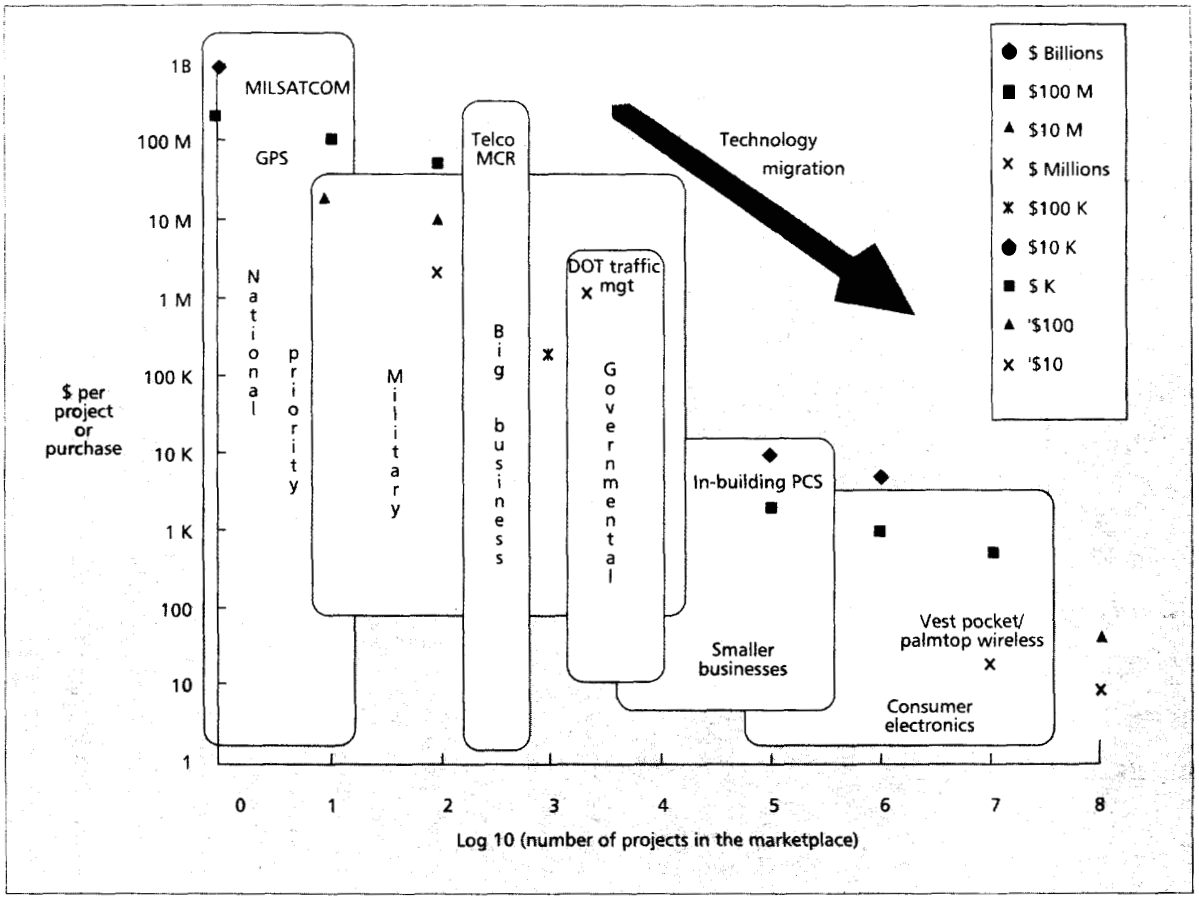
Software radios ideally place all IF, baseband, bitstream and source processing in a single processor. The assessment of the feasibility of the software radio centers on comparing estimated demand to the capacities of the available processors. Implementations back off from the ideal single-CPU implementation where driven to do so by processor, memory, or interconnect technology limitations, or to achieve cost advantages (e.g., of off-loading filtering to a Harris chip so that the DSP is less expensive). Software tasks are then structured into managed objects designed to run on any DSP or CPU with access to the data and sufficient processing capacity.

Open Architecture Software Tools Remain Problematic

Figure 7 shows how the time criticality of performance varies considerably with functionality. No signal processing environments in existence fully span the range from hard real-time IF processing through off-line support. Each tool and environment on the market excels in one or more aspects of the required support, although a few integrated environments approach this ideal. UNIX works well as both VME host and workstation host in laboratory systems. Portable VME systems often use a DOS/Windows laptop for user interface and display in lieu of a workstation, with a DOS/EISA or UNIX/VME signal processing configuration. SPOX and Harris Corporation's



■ **Figure 7.** *Software tools span the function space, but tool integration is far from seamless.*



■ **Figure 8.** *Economics of project size versus number of projects drive software radio technology migration.*

real-time UNIX excel in near-real-time support. Software tools for heterogeneous distributed processing are developing rapidly, but lack extended precision arithmetic and standardized application-to-application interfaces.

The Signal Processing Workbench (SPW) and COSSAP environments each span a wide range from analysis and design through VHDL code generation. These environments include block diagram languages that allow the designer to express processing in a signal flow representation. Block diagram languages are useful for concept development. The compilers can also map these to an intermediate language (typically C) which can be compiled and downloaded to target DSP chips (with up to an order of magnitude of performance degradation compared to hand coded C). Real-time access to the DSP status and performance is relatively hard to come by, however. Mercury and UNISYS' Militarized Touchstone environments provide counter examples, offering real-time operating systems with integrated performance monitoring and debugging capabilities. Some operating systems require one to build specialized status reporting tasks to send data of interest to the workstation. This is not a big impediment on a small project, but on larger, more complex projects, the integrated environments have significant advantages in reduced development times.

Rehosting tools tend to be aligned by vendor with Vendor-X providing a tool that will download C code to Vendor-X's own DSP hardware. The Signal Processing Workbench (SPW), Alta Corp., is a notable counter-example, with a very complete set of rehosting tools including targeting for several popular DSP cores and translation to VHDL for rapid transition to product hardware. Code libraries of such tool sets will evolve to standards-based suites of managed objets as object resource broker technlogy becomes more widely implemented in DSP tools.

Economics

Figure 8 shows the rough order of magnitude relationship between size of a project (or purchase) and the number of such projects (or potential purchases) in the marketplace. Software radios are now in the segment of the market dominated by the military, big business (e.g., the telephone and wireless service providers) and governments. The companion applications articles describe projects with costs on the order of a few million dollars to more than ten million, for example. Over time, the software radio will continue to move down and to the right as the size, power and cost of general purpose DSP chips, A/D and D/A converters and related interconnect and memory allow. In addition, object management software technology applied to software radios offers powerful

Software embodiment of traditionally analog and digital hardware functions opens up new levels of service quality and channel access flexibility.

tools for managing the increased complexity of emerging radio network standards within affordable acquisition and maintenance budgets. Within a few hardware generations, software radios will be in the vest-pocket and palmtop, providing the seamless communications services we aspire to.

Conclusions

Software embodiment of traditionally analog and digital hardware functions opens up new levels of service quality and channel access flexibility. In applications where access to multiple bands with multiple radio access modes is a necessity, the software radio can reduce hardware size, weight and power through fewer radio units. The need for such improvement appeared as chronic "interoperability" problems in military applications over a decade ago. Software radios address military issues as discussed in the companion article in this issue on Speakeasy, the military software radio. Software radios are also useful in introducing new channel access modes into bands where established modes must be accommodated for a number of years. As PCS and satellite mobile applications proliferate, the commercial sector is encountering continuing pressure for the increased flexibility and seamlessness of software radios. The software radio is a powerful architecture framework that helps us deliver such advanced radio services in a way that leverages the economics of contemporary microelectronics and software technologies.

References

[1] Bensley *et al.*, Introduction to Parallel Supercomputing, The MITRE Corporation, Bedford, MA, 1988.
[2] Pentek Corporation, (Laurel, MD: SPSI, 1994).
[3] The Common Object Resource Broker: Architecture and Specification, OMG 91.12.1, (Maynard, MA: Digital Equipment Corp. *et al.*, 1992).
[4] Kondo, Matsuo and Suzuki, "A Software-Defined Architecture Concept for Telecommunication Information Systems," ICC '94, (NY: IEEE Press, 1994).
[5] Voruganti, "A Global Network Management Framework for the 90's," ICC 94, (NY: IEEE Press, 1994), p. 1094.
[6] Celestino and Claude, "Building a Platform Support Environment for TMN Applications", ICC 94, (NY: IEEE Press, 1994) p.1847.
[7] Mitola "Software Radios: Survey, Critical Evaluation and Future Directions," Proceedings of the National Telesystems Conference, (NY: IEEE Press, May 1992).
[8] Kant, *Introduction to Computer System Performance Evaluation*, (NY: McGraw Hill, 1992).
[9] Collins and Tebbs, *Real Time Systems*, (UK: McGraw Hill, 1977).
[10] Kleinrock, *Queueing Systems*, vol. II, (NY: Macmillan, 1982).
[11] Mouly and Pautet, *The GSM System for Mobile Communications*, (France: Palaiseau, 1992).
[12] Ziemer and Petersen, *Digital Communications and Spread Spectrum Systems*, (NY: Macmillan, 1985), p. 694.

Biography

JOE MITOLA is a consulting scientist at the MITRE Corporation, McLean, Virginia, a not-for-profit corporation that operates federally funded research and development centers for the U.S. Government. His research interests include quantitative methods in systems engineering, radio technology, and information theory. He advises the U.S. Government on large-scale systems architectures, telecommunications, and information processing technologies. Prior to joining MITRE in 1993, he was chief scientist of Electronic Systems, E-Systems Melpar, culminating more than a decade of positions of increasing responsibility. Earlier in his career, he worked for the Harris Corporation, Advanced Decision Systesms, and ITT. He began his career at the U.S. Department of Defense in 1967. He received a B.S.E.E. from Northeastern University in 1972 and an M.S.E. from Johns Hopkins University in 1974. In addition, he passed the Ph.D. qualifying examinations in Information Processing (Articificial Interlligence) and Systems Theory at the University of Maryland in 1985. His e-mail address is: jmitola@mitre.org.