A Scalable Open-Source Digital Video System for Launch Range Operations

Michael M. Gorlick and John Georgas
The Aerospace Corporation
El Segundo, California
`gorlick@aero.org`, `georgas@aero.org`

I. Introduction

Raging incrementalism [1] is a systems engineering methodology designed to capitalize on the force of *hyperexponential change*, the ongoing acceleration of overall technical progress. Numerous pivotal technologies are not only improving at an exponential pace, but the pace itself is accelerating [2], yielding hyperexponential improvement overall.

In the large, the rate of technical progress is doubling every decade. Taking the 1990s as a nominal benchmark decade of technical progress then the first 25 years of the 21st century shall see the equivalent of 100 years of 1990s-like progress. By 2100 we will have experienced a period of change comparable to **20,000** years of unending 1990s technical progress.

Change of this pace, sweep, and magnitude calls into question all of the accepted canons of system engineering and development. In response, raging incrementalism leverages commodity hardware, open-source software, stateless, protocol-centric transactions drawn from the design principles underlying the web [3], and peer-to-peer architectures to create scalable infrastructure that can easily absorb massive, ongoing change.

It is one thing to propose or discuss a "style" or methodology of system engineering and quite another to put it to practice.[1] To better understand the utility and consequences of raging incrementalism we are constructing a variety of complex applications, including two launch range services, a countdown and timing infrastructure (RACE, the RAnge Countdown Experiment) and a wide-area digital video system (RAVE, the RAnge Video Experiment). This paper offers a review of the development, architecture, and construction of RAVE.

II. RAVE Requirements

Lompoc, California is home to Vandenberg Air Force Base and the Western Launch Range, approximately 100,000 acres of space launch pads and missile test silos, and the only launch site in the United States suitable for the polar orbital launches required for many satellite weather and reconnaissance missions. The Western Launch Range, like its sister the Eastern Launch Range at Cape Canaveral, Florida, contains many hundreds of analog video cameras. These cameras are essential tools for range security, launch preparation, launch monitoring, launch vehicle and pad engineering, and post-mortem analysis. On the Western Range this multitude of analog video cameras feeds an enormous 1980s era video switch that, though a miracle of engineering in its day, is now obsolete and increasingly difficult to repair. In addition the video system relies upon a dedicated and aging cable plant that is falling into disrepair and is expensive to maintain.

[1]Steve Crocker, one of the founding fathers of the modern Internet, once quipped "The difference between theory and practice is much smaller in theory than it is in practice."

RAVE is a prototype replacement for the Western Range video infrastructure whose goal is an all-digital video system, based upon modern open networking standards, that is constructed entirely from commodity hardware and open-source software. The vast bulk of the video cameras on both ranges are black and white security cameras; it is these cameras that were our initial "targets" for replacement. Loosely stated, we required that RAVE must both match or exceed the current (or near-future) range video system:

- Full color, 320×240 video at a minimum of 15 frames per second
- Standards-compliant, high-quality, video compression throughout
- Scaling to hundreds of individual cameras

while providing significant new functionality:

- Remote network control of all cameras (positioning, focus, on/off, color balance, frame rate, and the like)
- Arbitrary remote network video clients ranging from conventional desktop hosts to terabyte-capacity video archives
- Arbitrary video switching; that is, every video source is accessible to every video client
- Precision timestamps, synchronized to a range-wide timebase, for each and every video frame

III. Commodity Hardware

Raging incrementalism recommends (as do others) that system architects, whenever feasible, take full advantage of commodity hardware as commodity components reduce costs, simplify maintenance, and reduce provisioning to a web click. In this spirit we selected Firewire video cameras as our starting point, as their control interface is rigorously specified [4], they are widely available [5], offer price/performance points suitable for almost all applications, and Firewire interfaces are now commonplace on commodity computing platforms.

Raging incrementalism also suggests that where possible, specialized hardware be eschewed in favor of software; in other words, use bits—not iron. This implies that the real-time video encoding and compression required for the range be done entirely in software rather than relying on proprietary hardware. Further, since all RAVE cameras are network-accessible, it is easy and inexpensive to remotely and automatically replace a software-based video codec with an improved version (or an altogether different codec). In contrast, replacing hardware codecs in cameras scattered over 100,000 acres of range real estate is time-consuming, expensive, and guarantees obsolescence. As codecs improve in performance (and hyperexponential change guarantees that they will) a software solution ensures that the RAVE cameras will remain current longer than they might otherwise.

Finally, raging incrementalism encourages the encapsulation of system functions into *bricks*—hosts dedicated to a single function or a small, strongly related family of functions. To illustrate, a RAVE *camera brick* is just a dedicated, small form-factor, commodity computer (running Linux) connected to a Firewire video camera.

IV. Open-Source Software

Digital video is a bewildering cacophony of standards and conventions, and that diversity and complexity is reflected in the open-source offerings. For the sake of brevity we list here only the critical open-source packages and applications that appear in RAVE:

- Debian Linux and FreeBSD, two of the premier open-source operating systems
- `libdc1394`, a C library for the control of IIDC-compliant Firewire video cameras
- `spook`, a video broadcaster and camera control application
- `xViD`, a high-performance MPEG-4 codec
- `mencoder`, a command-line MPEG-4 encoder and multimedia container generator built atop `xViD`
- `Darwin Streaming Server`, the streaming media server from Apple
- `mplayer`, a well-known video playback tool that runs under Linux, Mac OS X, and Windows
- `MySQL`, an open-source relational data base
- `C`, `C++`, and `Python`
- `Pyrex`, a wrapping and interface tool for integrating C libraries into Python
- `Live.com RTP/RTSP` library for the management of RTP/RTSP media streams
- `Mozilla Firebird` web browser as the user interface for control of cameras and streaming servers

In addition we rely upon the open protocols RTP, RTSP, HTTP, TCP, and UDP.

V. RAVE Architecture

The structure of the RAVE system is illustrated in Figure 1. *Camera bricks* generate MPEG-4 digital video streams for eventual delivery to clients and applications. *Streaming bricks*,

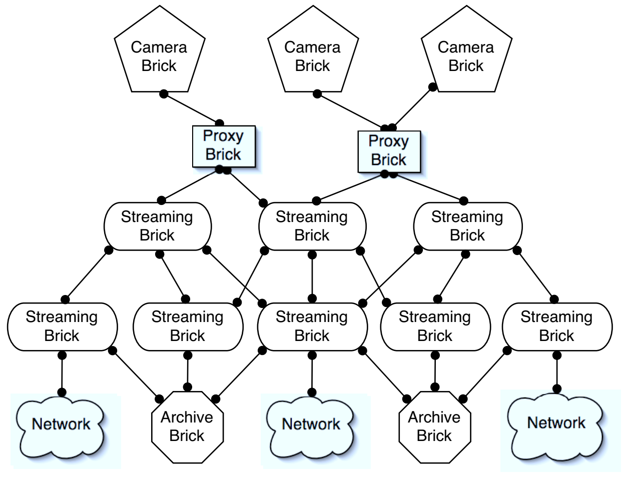

Fig. 1. Notional Architecture of RAVE.

based upon the Darwin Streaming Server (DSS), deliver multiple camera streams to multiple clients on demand. *Proxy bricks* act as protocol bridges between cameras and streaming bricks and insulate cameras from the protocol details of communicating with instances of DSS. *Archive bricks* are terabyte-scale video archives that subscribe to and record, for archival purposes, video streams of interest. Finally, other video clients residing within *network clouds* subscribe, via the streaming bricks, to video streams of interest.

VI. Rapid Development

Raging incrementalism emphasizes the simultaneous development and continual integration of "found" open-source components. Under these conditions, the architecture must be "naturalistic," that is, the architecture is shaped "bottom-up" to accommodate the available components. The RAVE architecture and components were developed simultaneously and the four principal RAVE bricks—camera, proxy, streaming server, and archive—were developed concurrently and integrated over approximately a 6-week period. Since each distinct kind of brick is protocol-centric (hence separable to a large degree from the remainder of the system) it was comparatively easy to test each brick in isolation and then integrate it into RAVE.

VII. Lessons Learned

Integration is a distinct skill from either design or programming, and in a world of hyperexponential change it is the sport of kings. It requires a deep understanding of the domain, knowledge of multiple programming languages, the ability to quickly review and extract the salient features of multiple, disparate components, and a strong sense of architectural styles and esthetics.

Digital video is at the bleeding edge of almost everything Linux, from kernel-level device drivers, to middleware, to applications. Evaluating and testing various open-source components often required a "magic blend" of kernel modules and libraries that were specific to, and different for, each component. This complicated development considerably, and on several occasions we were forced to rebuild a Linux box from scratch when the tangle of modules and libraries produced irreconcilable conflicts. A set of version-controlled operating system images would clearly be helpful in such circumstances.

While integration (rather than "clean slate" development) may be complex and difficult, it is the best (and only) hope for pacing hyperexponential change.

VIII. Conclusion

Large-scale digital video is nontrivial and is a challenging element of range infrastructure and operations. RAVE, constructed entirely from commodity hardware and open-source software components in a short period of time, is an excellent initial demonstration of the utility of raging incrementalism. Future experiments will address scaling the prototype and testing the resilience of the architecture in response to changes in components and the addition of new functions.

IX. Acknowledgments

This work was sponsored by the Research and Development Program Office of The Aerospace Corporation.

References

[1] Michael Gorlick, "Raging Incrementalism—System Engineering for Continuous Change," *Proceedings of the 2004 Ground Stations Architecture Workshop*, Manhattan Beach, California, March 2004.

[2] Ray Kurzweil, *The Law of Accelerating Returns*, March 7, 2001, `www.kurzweilai.net/articles/art0134.html?printable=1`.

[3] Roy T. Fielding and Richard N. Taylor, "Principled Design of the Modern Web Architecture," ACM Transactions on Internet Technology, 2(2), May 2002, pp. 115-150.

[4] "IIDC 1394-based Digital Camera Specification," Version 1.30, July 25, 2000, 1394 Trade Association, Santa Clara, California.

[5] "The IEEE 1394 Digital Camera List," `http://www.tele.ucl.ac.be/PEOPLE/DOUXCHAMPS/ieee1394/cameras/`. A detailed listing of over 150 IIDC-compliant Firewire cameras ranging in price from $17 to $15,000.