



Large-Scale, High-Bandwidth Video and Robotic Camera Control With Java™ Technology

java.sun.com/javaone/sf

Shad J. Aumann, Senior Software Engineer
Robert A. Cross, Ph.D., Chief Scientist
LiveWave, Inc.

<http://www.livewave.com/>



Scalability Across the Feature Set

Enterprise Approach to Video and Device Control

Examine how FirstView[®] delivers high-fidelity video and interactive device control using Java[™] technology.

Session Agenda

High-level System Design

Robotic Device and Sensor Control

Video Capture, Encoding, and Distribution

Video and Sensor Control Clients

Deployment Architecture

Live Demonstration—Drive the Cameras

Session Agenda

High-level System Design

Robotic Device and Sensor Control

Video Capture, Encoding, and Distribution

Video and Sensor Control Clients

Deployment Architecture

Live Demonstration—Drive the Cameras

Distributed Application

The collective

- Shared database
 - Application settings
 - Application state
- Linear scalability
 - One or more roles per system
 - 8 channels per encoder
 - 64 channels per repeater
 - System roles can be reassigned
- Limiting factors
 - Network bandwidth
 - Human operator

Data Rate Diversity

Throughput, latency, and delivery requirements

- Video
 - High throughput, low latency
 - Delivery consistency can be variable
- Control messages
 - Low throughput, low latency
 - Delivery, if possible
 - State machine helps the robot when delivery fails
- Sensor events
 - Low throughput, moderate latency
 - Delivery must go through

User Role Diversity

Orthogonal market segments

- Broadcasters
 - Small number of cameras
 - Comprehensive camera settings
 - Precise control and playback of movements
 - Preview/program workflow
- Surveillance
 - Large number of cameras
 - Integration of video with other sensors
 - Hierarchical roles and permissions
 - Event oriented workflow

Device Diversity

Protocols and communication interfaces

- Serial devices
 - Robotic cameras
 - Robotic mounting heads
 - Environmental sensors
 - USB input devices
- Data network devices
 - Serial port servers
 - Video servers
- Other integrated devices
 - DVR
 - Video encoders

Session Agenda

High-level System Design

Robotic Device and Sensor Control

Video Capture, Encoding, and Distribution

Video and Sensor Control Clients

Deployment Architecture

Live Demonstration—Drive the Cameras

Sensor Control With Java Communication Technology

Two communication problems, one interface

- Hide device diversity from end users
 - Focus on interaction with the system
- Direct connection to serial port
 - javax.comm supports RS-232
- Remote connection via network port servers
 - RS-232, RS-422, RS-485
- Implementation
 - Communicator interface hides direct vs. remote
 - Device-specific control protocol

Soft Real-Time Robotic Device Control

Performance guidelines

- Eyes on the target metaphor
- Most robotic devices are synchronous
 - State machine: simple but required
- Device-specific issues
 - Velocity-based control vs. absolute positioning
 - Start/stop bookends vs. continual commands
 - Pan/tilt head + camera: must control as if one unit
- Feedback is visual, round-trip is key
 - Interaction requires latency ≤ 100 milliseconds
 - Slower communications motivate “presets”

Interfacing With Other Sensor Types

Chemical, radiation, motion, laser, contacts, etc.

- Specific devices
 - Chemical weapon detectors via JMS™ API
 - Serial contact closures
 - Radiation detection via CCD (astronomy algorithms)
 - Motion detection in the video stream
 - Laser rangefinder
- Often asynchronous communication
- Integration requires careful design
 - Not just flinging bytes
 - Camera + laser rangefinder
+ absolute position pan/tilt head = target location

Soft Real-Time Image Processing

Applications of Java Advanced Imaging

- Simple motion detection
 - Establish a baseline
 - Detect and interpret changes
 - Straightforward with good scalability
- “Left object” detection
 - Motion detection in reverse
- JAI performance considerations
 - Evaluate operations as late as possible
 - Native code, hardware acceleration possible

Soft Real-Time Image Processing

JAI operations supporting motion detection

- `javax.media.jai.operator` package
 - JAI operation descriptors
- **JPEG**
 - Decoding MJPEG video stream
- **SubsampleAverage**
 - Reduce frame size
- **DivideByConst**
 - Rolling mean of previous images
- **Absolute and Subtract**
 - Difference from mean

Soft Real-Time Image Processing

Applications of Java Advanced Imaging

- Motion detection in recorded stream
 - Same algorithm, different source
- “Left object” detection in high-traffic area
 - Long baseline image eliminates most traffic
 - Rolling baseline image is different from expected
 - Object fits parameters: possible alert
 - Not magic: people are opaque
- FirstView is always a collective
 - Many simple algorithms, all with a vote
 - More assets on target = greater certainty
 - Humans are always the most valuable resource

Session Agenda

High-level System Design

Robotic Device and Sensor Control

Video Capture, Encoding, and Distribution

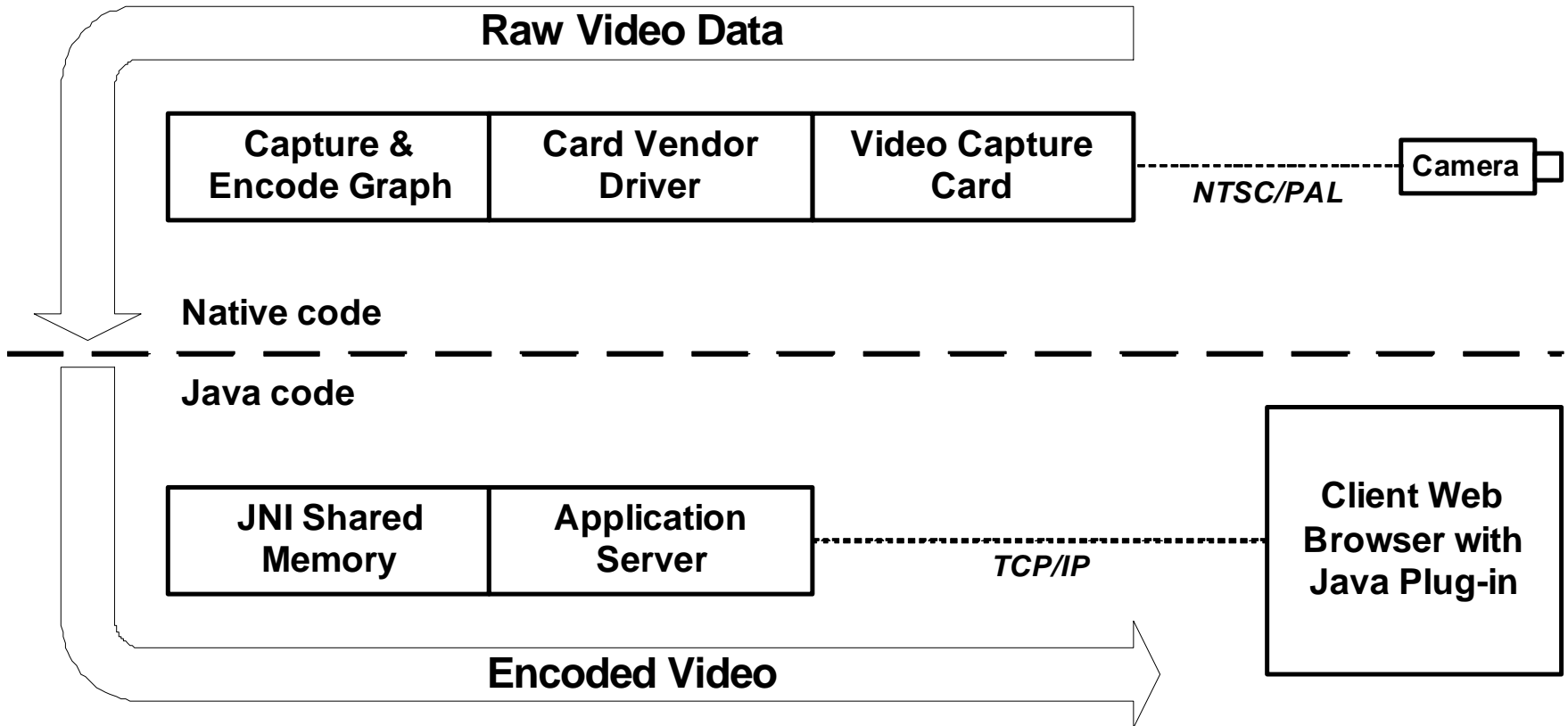
Video and Sensor Control Clients

Deployment Architecture

Live Demonstration—Drive the Cameras

Video Data Flow

Video capture, encoding, and distribution



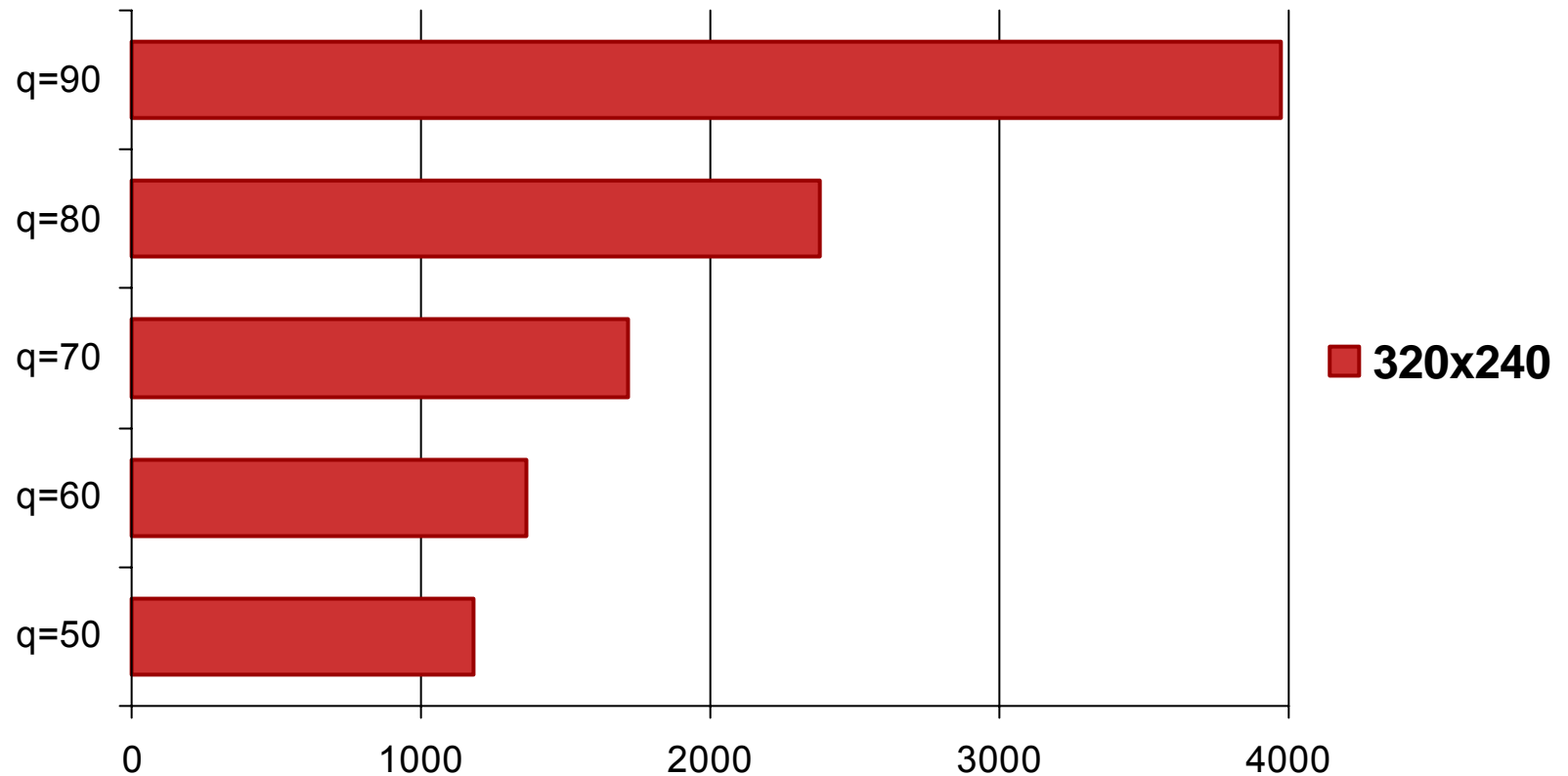
Video Data Flow

Video capture, encoding, and distribution

- Java Media Framework
- Custom C++ application
 - Enumerates video input devices
 - Creates a capture and codec filter graph
 - Plays the graph
- Java video rendering implementation
 - Network sockets
 - Shared memory
- Streaming to clients

Video Data Rates With MJPEG Codec

Kilobits/second at 30fps with varying quality



Source: FirstView® application logs

Session Agenda

High-level System Design

Robotic Device and Sensor Control

Video Capture, Encoding, and Distribution

Video and Sensor Control Clients

Deployment Architecture

Live Demonstration—Drive the Cameras

Web User Interface

Java Servlets and JavaServer Pages™ technology

- Role-based user interface
 - Hierarchical
 - Dynamic
- Broadcasting
 - Detailed camera settings
- Surveillance
 - Multiple cameras
 - Asynchronous behaviour
- Enumeration of device characteristics
 - Video switchers
 - Fixed vs. pan/tilt cameras

Web User Interface

USB joystick in web applets

- Native joystick driver
- 3-axis support
 - Up/down, left/right, twisting
 - Tilts, pans, zooming
 - Buttons recall preset camera positions
- Virtual joystick devices
 - Digital tablet
 - Touch screen

Web User Interface

Solving the native library problem

- Code signing
- Native library use in applets
 - Different JAR files use separate class loaders
 - Sharing across class loaders is forbidden
- Super JAR
 - Multiple applets in one JAR
- Native library renaming scheme
 - Signed applet manages native library

Java Plug-in and JavaScript™ Technology Interaction

Calling JavaScript programming language from an applet

- LiveConnect
 - `netscape.javascript.JSObject`
- `AppletContext.showDocument()`
 - javascript: URL scheme
 - Browser plays page transition sound
- Common DOM API
 - `DOMService`

Java Plug-in and JavaScript Technology Interaction

Calling an applet from JavaScript programming language

- Sample bugs in 1.4.2 releases
 - “Java->JS crashes IE”
 - “JavaScript event handling becomes asynchronous”
- JavaScript stack can overflow
 - Must prevent calling into applets too quickly
 - Do not forget about the network
- Smooth control requires event throttling
 - Last event wins
 - Outgoing priority queue
 - Limit events per unit time

Throttling JavaScript Events

Event delay and cancellation

```
var theApplet = document.getElementById("control");  
var currentZoomValue = 0;  
var guiTimeout;
```

```
function setZoom(zoomValue){  
    // Cancels waiting call to appletSetZoom()  
    clearTimeout(guiTimeout);  
  
    currentZoomValue = toDecimals(zoomValue, 0);  
  
    // Calls appletSetZoom() after 25ms delay  
    guiTimeout = setTimeout("appletSetZoom()", 25);  
}
```

```
function appletSetZoom(){  
    // Calls the applet's public method setZoom()  
    theApplet.setZoom(currentZoomValue);  
}
```

Displaying Video

Video in an applet

- Decoding the video stream
 - Snapshot feature
 - Annotation feature
 - Integration into collective-wide features
- Graphics toolkit decoding
- JAI decoding
- JMF decoding
- Custom component with signed applet

Session Agenda

High-level System Design

Robotic Device and Sensor Control

Video Capture, Encoding, and Distribution

Video and Sensor Control Clients

Deployment Architecture

Live Demonstration—Drive the Cameras

Deployment Architecture

Software components

- Server
 - Microsoft Windows XP Professional
 - Java™ 2 Platform, Standard Edition 1.4.2
 - Caucho Resin 2.1.x
 - JDBC™ API compatible RDBMS
- Client
 - PC and handheld with web browser
 - Java technology implementation
 - Java Plug-in
 - PersonalJava
 - MSJVM
 - Mouse, USB joystick, touch screen

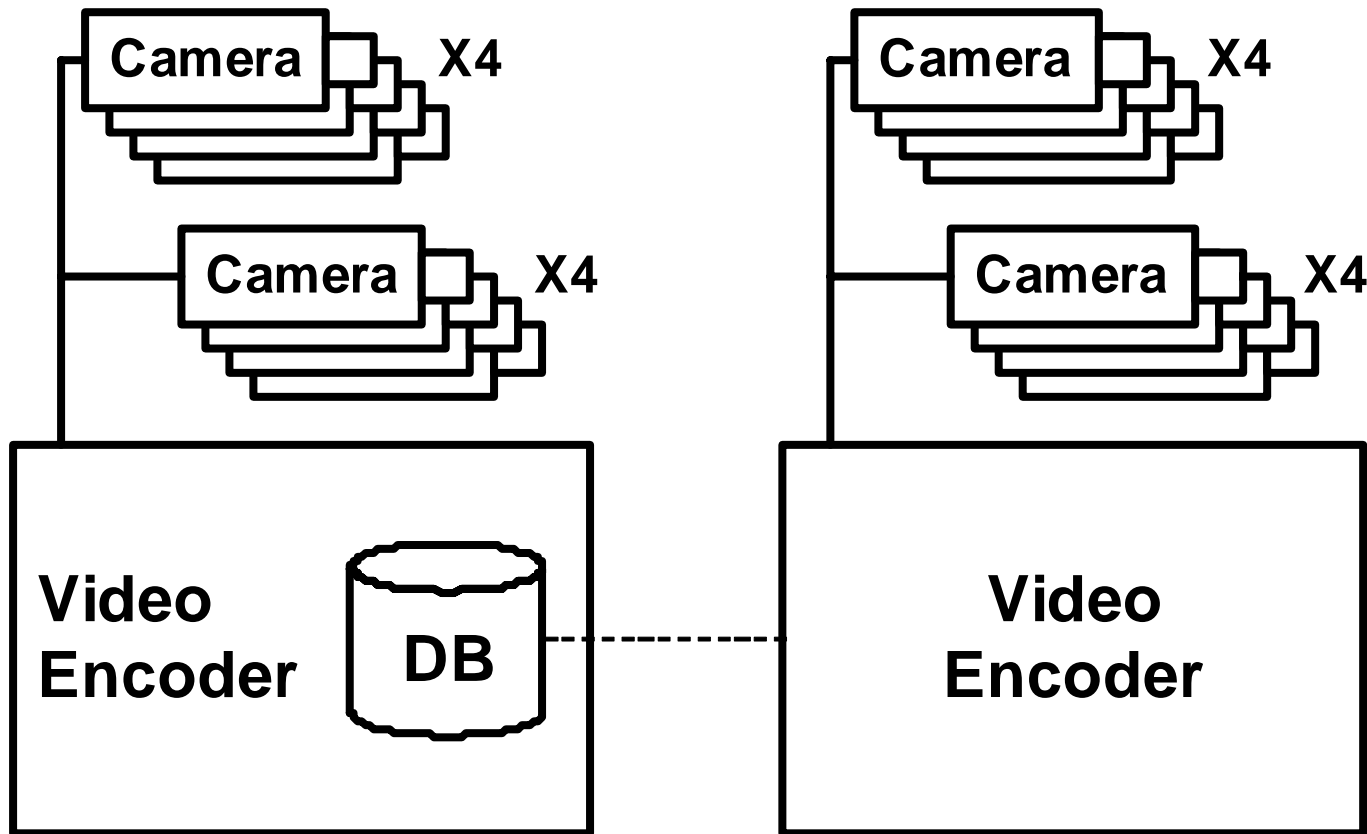
Deployment Architecture

Hardware components

- Typical configuration
 - 1U or 2U rack mountable form factor
 - Single 3GHz CPU
 - 1GB RAM
 - 1000-baseT Ethernet
 - 2X quad-input video capture cards
- DVR configuration
 - 4U rack mountable form factor
 - Dual CPUs
 - 16X 250GB SATA RAID

Deployment Architecture

Collective of up to 16 channels



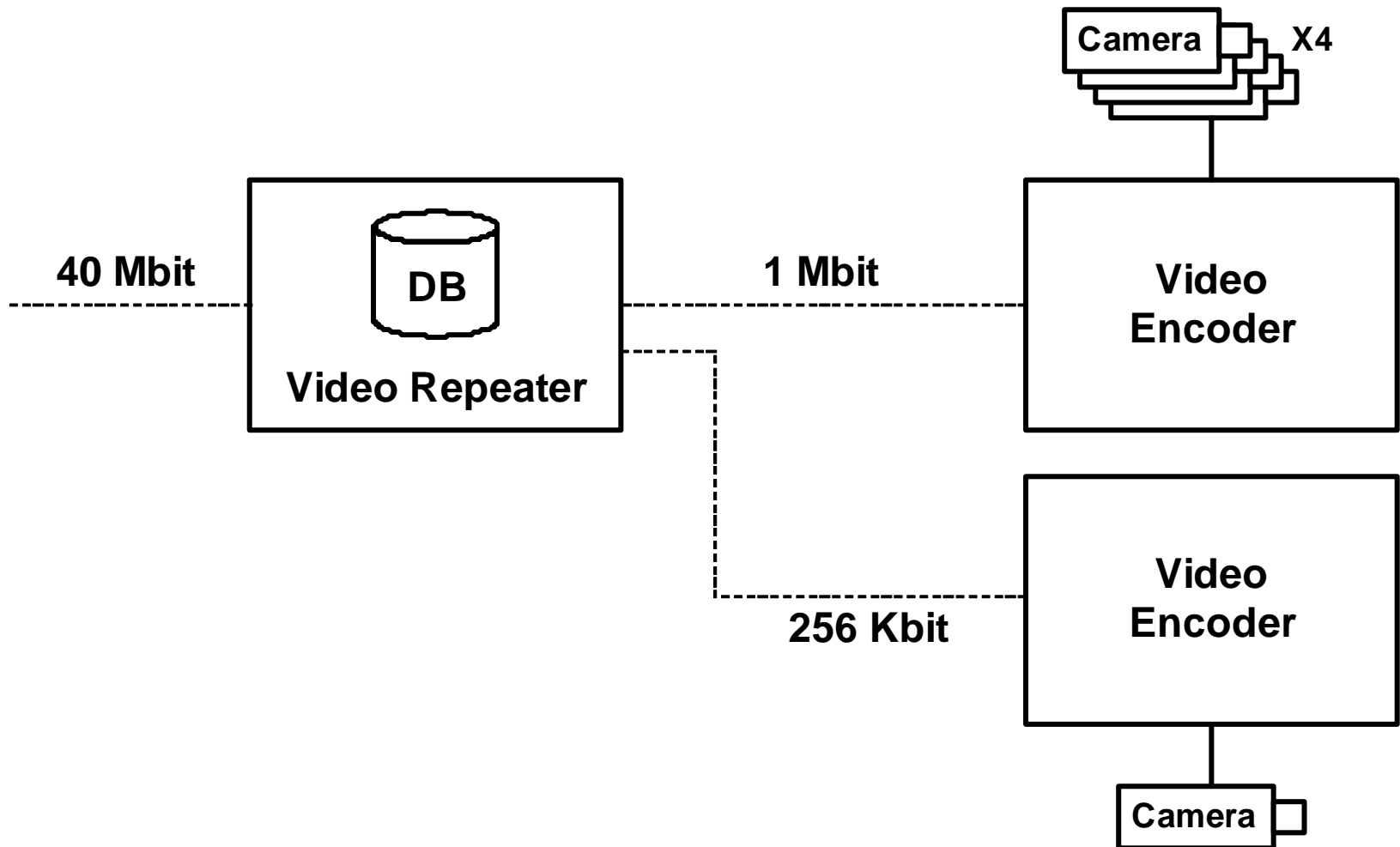
Hourglass-shaped Architecture

Minimizing end-to-end network traffic

- Network cannot handle aggregate bandwidth
 - 200 cameras, 15fps, q=50
 - 200Mbit/second aggregate
- Video repeaters localize traffic
 - Low count of connections to encoders
 - Higher connection counts near clients
- Video distribution concept of operations
 - Minimize “end-runs”
 - Provide access as close to clients as possible

Deployment Architecture

Concentrating and segmenting bandwidth



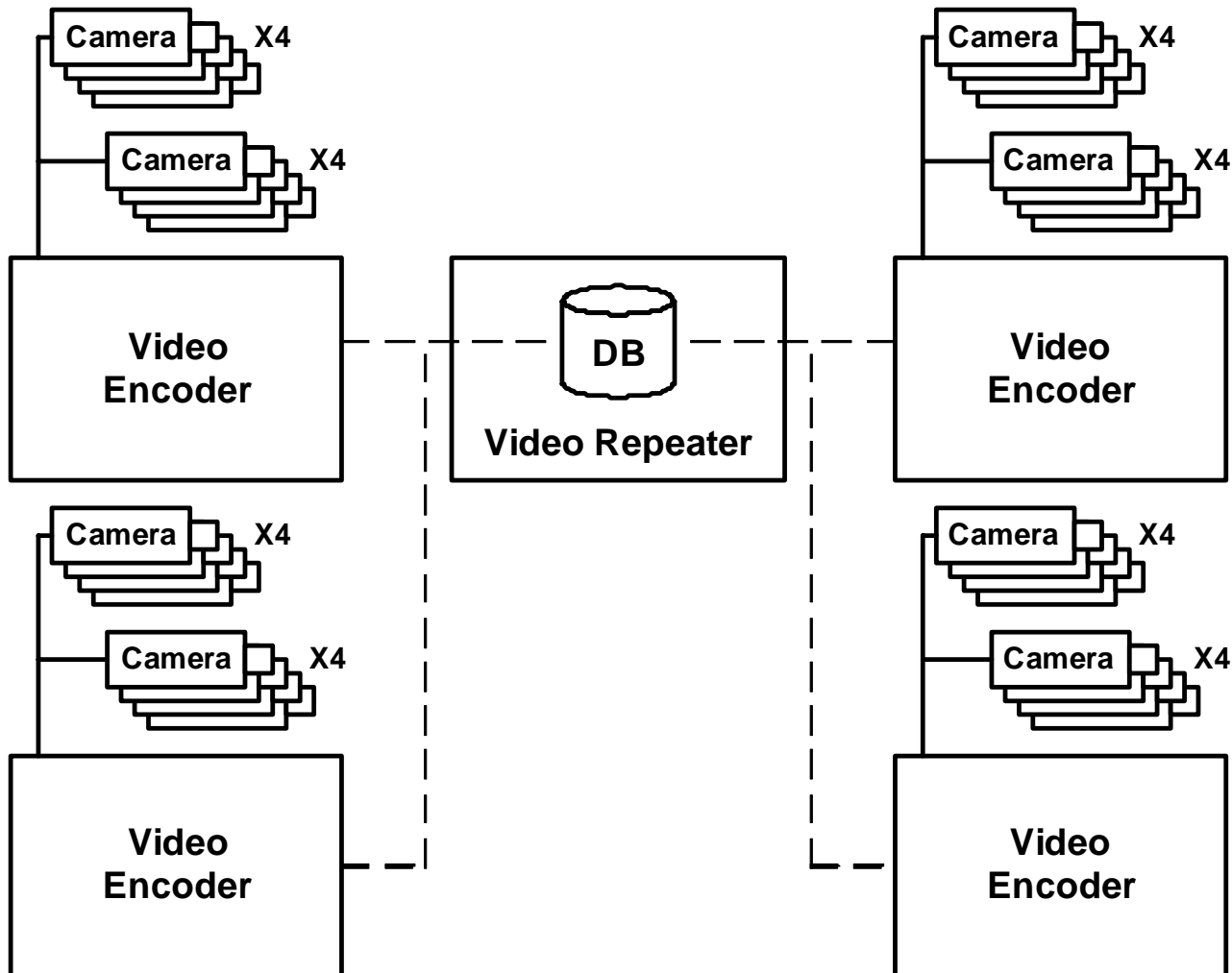
Real-world Deployment

Washington, D.C. Metro (WMATA)

- 54 servers in 20 subway stations
 - 41 video encoders
 - 13 video repeaters
- 360 channels
 - 45 are PTZ cameras
 - Others are fixed cameras or repeated streams
- Channel frame rates
 - Up to 7200 frames per second
 - Minimum of 15 FPS per channel totals to 5400

Deployment Architecture

Collective of up to 32 channels



Real-world Deployment

Federal protective services

- Several buildings in the Boston area
- Nationwide hierarchical control
 - Coordination with local first responders
 - City fire and police departments
 - Federal agencies
- Potential hierarchy
 - City level
 - State level
 - Regional level
 - Nationwide level

Session Agenda

High-level System Design

Robotic Device and Sensor Control

Video Capture, Encoding, and Distribution

Video and Sensor Control Clients

Deployment Architecture

Live Demonstration—Drive the Cameras

Demo

Drive the Demo Cameras
802.11b/g SSID “democams”



Summary

- High-fidelity video
- Interactive control across the collective
- Remote management
- Dividing and conquering across the collective
- Broadly leveraging Java technology
- Deploying to a city near you

For More Information

- Public demo cameras
 - <http://www.livewave.com/cambrowser/>
 - <http://dsc.discovery.com/cams/sharkvideo.html>
- Java products and technologies
 - Java Advanced Imaging
 - Java Comm API
 - Java Database Connectivity
 - Java Media Framework
 - Java Plug-in
 - Java Servlets
 - JavaServer Pages

Q&A

Shad J. Aumann

Robert A. Cross, Ph.D.



Large-Scale, High-Bandwidth Video and Robotic Camera Control with Java™ Technology

java.sun.com/javaone/sf

Shad J. Aumann, Senior Software Engineer
Robert A. Cross, Ph.D., Chief Scientist
LiveWave, Inc.

<http://www.livewave.com/>

