Toward a Common Infrastructure for Digital Media Middleware

Steven McCanne
University of California, Berkeley

MCWS ’97
February 1997

The Problem

Many independent pieces of networked multimedia technology:

- LBL/UCB MBone tools (vic, vat, wb)
- UCL tools (nte, rat, sdr)
- INRIA tools (ivs, FreePhone)
- Holfelder's MBone VCR
- UCB/Cornell CMT apps
- Vu-System
- Measurement/debugging tools (mtrace, rtpmon)

Very little leveraged across these projects...

Toward Common Infrastructure

Infrastructure is less and less a research issue...

Can we leverage each other’s work by adopting common architectures?

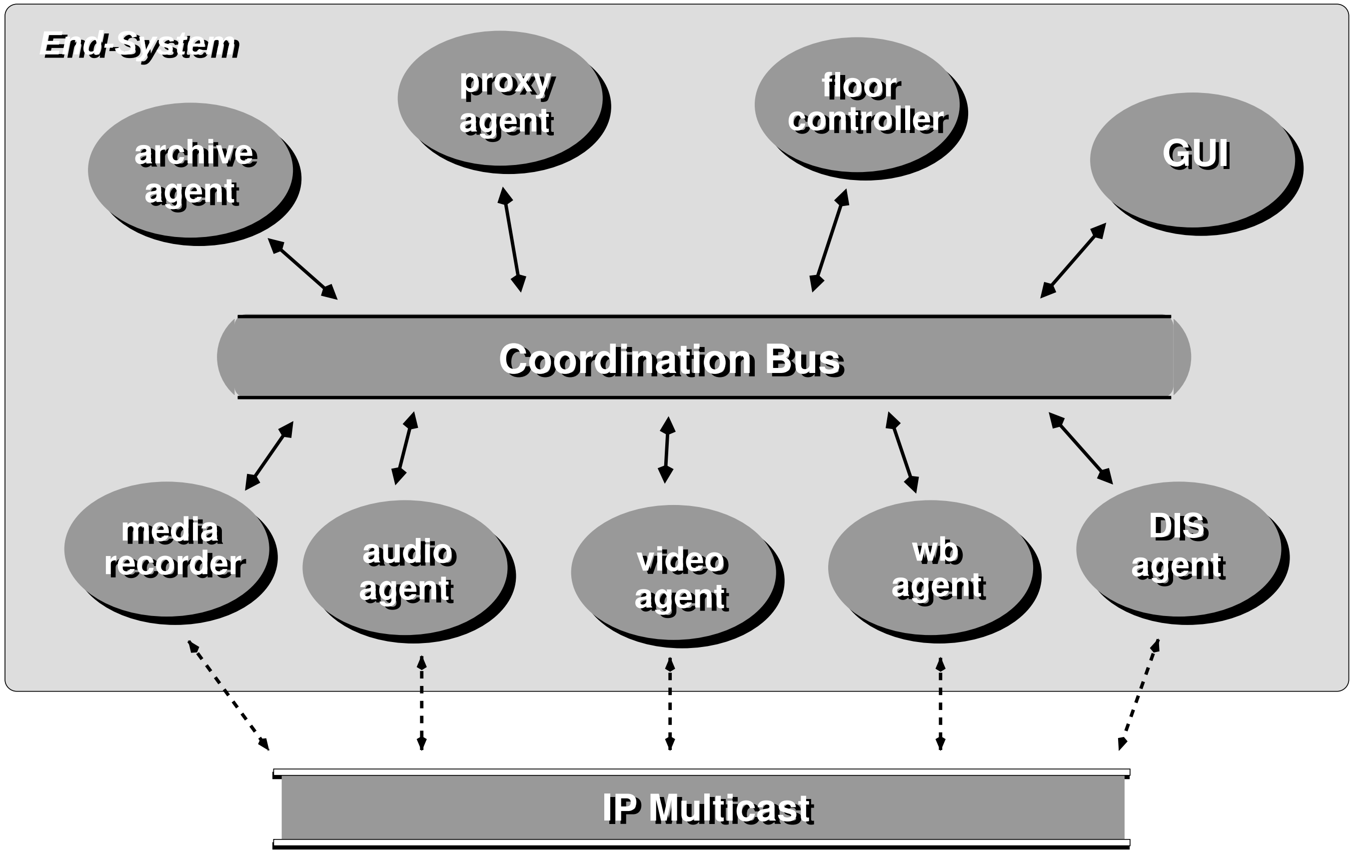
“The BSD of multimedia networking...”

Our two approaches:

- coarse-grained inter-agent APIs (“Coordination Bus”)
- fine-grained multimedia objects

Coordination Architecture

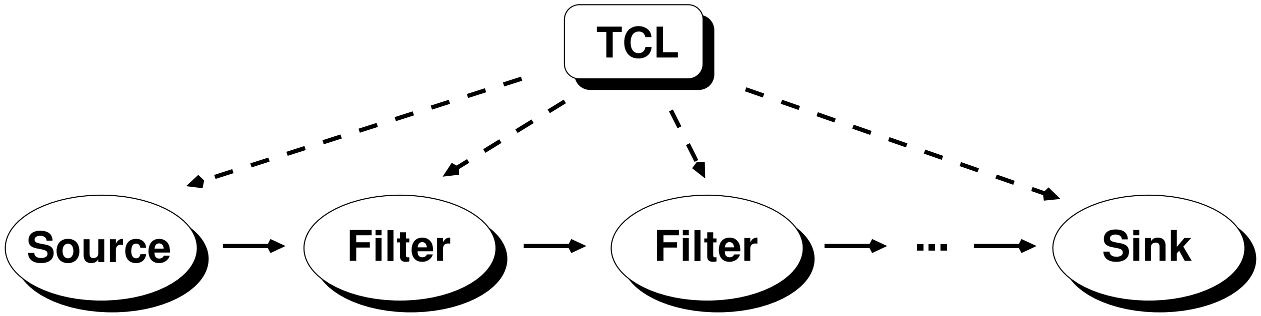
Large number of interdependent tools orchestrated over *Coordination Bus*:



Toolkit-based Agents

How to leverage commonalities across agents?

Agent-based toolkit for compositing digital media objects:

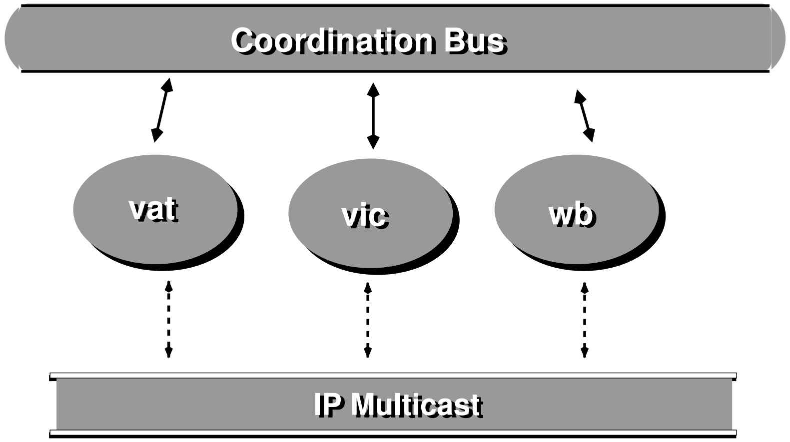


- Modules implemented in compiled language (e.g., C++)
- Configured, linked, and controlled in scripted language (e.g., Tcl)

This approach independently developed in the MBone tools, UCB Plateau's CMT, and MIT's Vu-System.

MBone Tools

What do the MBone tools offer?



+ preliminary support for coordination bus
+ robust, widely deployed, and readily available
- decomposition too coarse-grained (“composable tools”)
- ad hoc and unmanageable Tcl implementation

Continuous Media Toolkit (CMT)

+ well-developed architecture, documented API's
+ rich infrastructure for multimedia authoring
+ active project
- no inheritance
- implemented in C
- complexity often overloaded into objects rather than decomposed (e.g., JPEG decoder know about dithering and X windows)

Vu-System

+ fine-grained media objects
+ elegant, object-oriented Tcl extension
+ clean architecture: C++/Object-Tcl inheritance
- project inactive
- inadequate network support (no RTP, no multicast)
- custom UI toolkit limited
- Vu-System core uses primitive version of OTcl

Our Plan

Combine strengths of all previous approaches:

- immediately leverage MBone tools' code base and robustness
- retrofit Vu-System's elegant object model (Object Tcl or "OTcl")
- develop scaffolding between new toolkit and CMT
  (and a long-term collaboration strategy)

MASH Toolkit

- MASH Toolkit = collection of OTcl/C++ objects

- MASH Shell = Tcl interpreter + MASH objects

vic, vat, *mediaboard*, and new applications are simply scripts that are interpreted by the MASH shell.

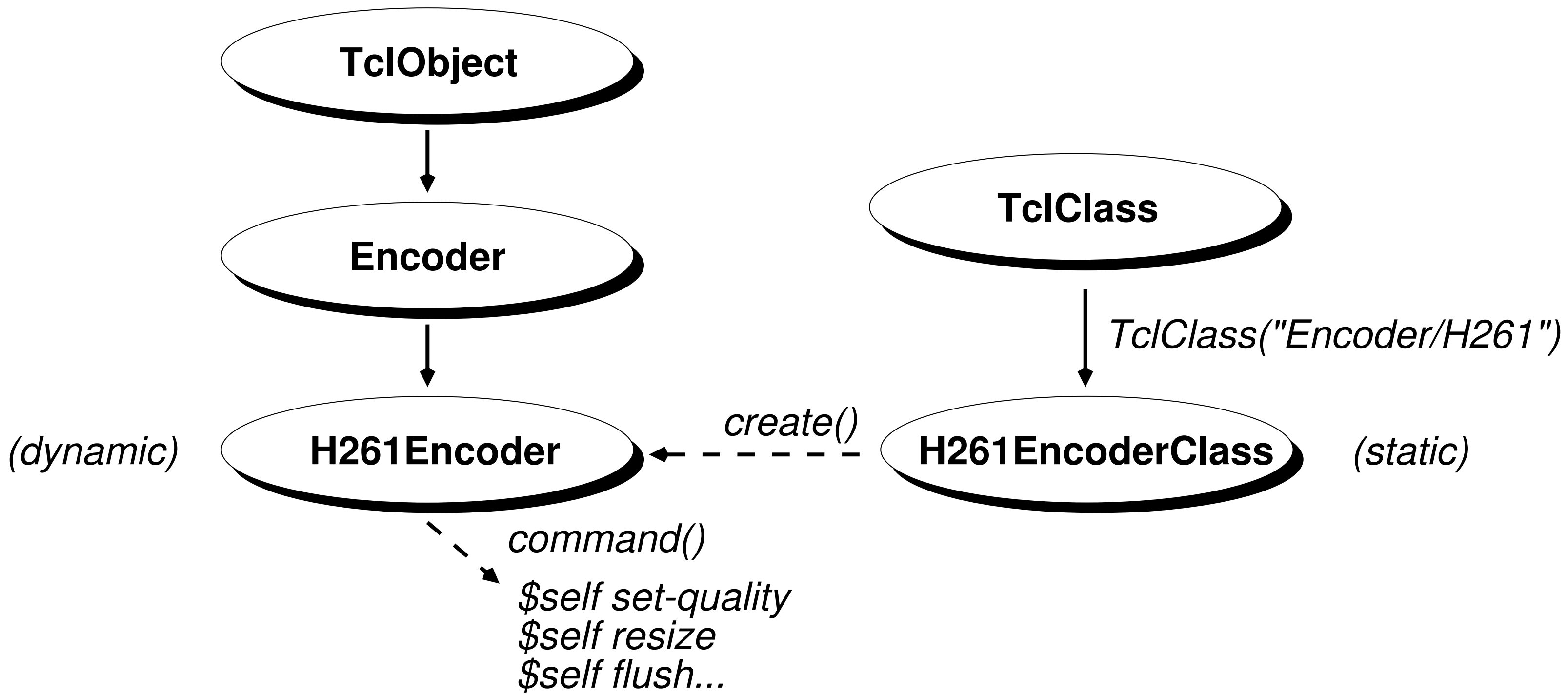
MASH Toolkit (cont'd)

Object Tcl has minimal support for C and none for C++.

MASH toolkit extends OTcl across C++:

- *new* and *delete*
- shadow objects
- TclClass
- TclObject

Example MASH Object



No edits to existing files!

Example MASH Object (cont'd)

Can define additional methods in OTcl:

```
Encoder/H261 instproc geometry {} {
        $self instvar width_ height_
        return [concat $width_ x $height_]
}
```

Define instance variables in either C++ or OTcl:

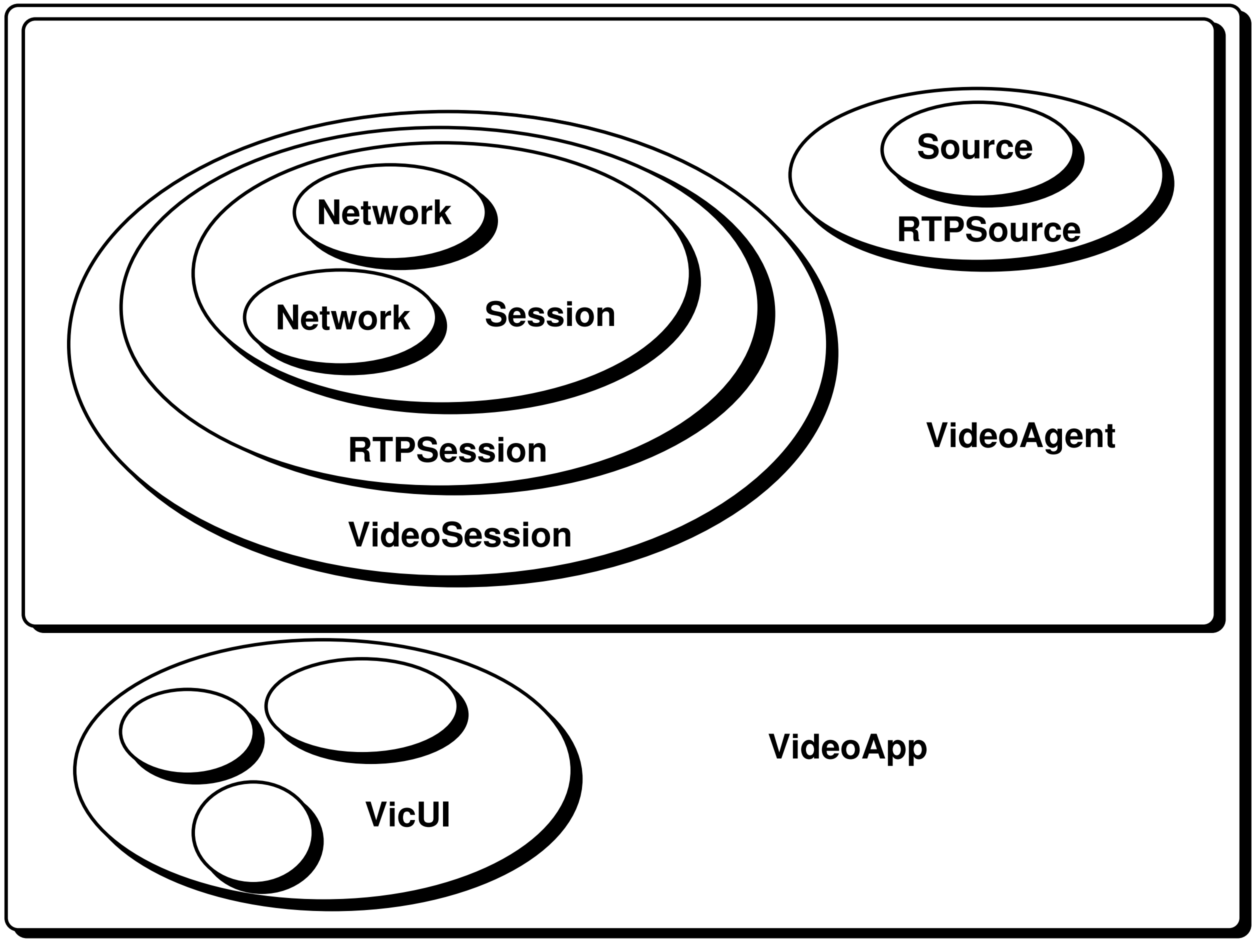
```
H261Encoder::H261Encoder()
{
        bind("width_", &width_);
        width_ = 320;
        ...
}
```

vs.

```
$self set width_ 320
```

bind() simply uses *Tcl_TraceVar*

Abstraction Hierarchy



MBone Tools into MASH

Factored MBone tools into OTcl classes to create MASH toolkit.

Separated user interface from bare mechanism:

- Simply imposed object-oriented organization on existing MBone tools Tcl scripts (with small changes to C++ API).
- Migrated all object creation/deletion into Tcl

Object Composition

Easy composition. To create vic-like application:

```
set agent [new VideoAgent ...]
set ui [new VideoUI ...]
$agent attach $ui
```

To create a combined A/V tool:

```
set audio [new AudioAgent ...]
set video [new VideoAgent ...]
set ui [new AudioVideoUI ...]
$audio attach $ui
$video attach $ui
```

Easy to “throw together” apps by leveraging toolkit components.

Summary

Described object-oriented digital media middleware (the MASH shell) that:

- is compatible with MBone tools (RTP & SRM),
- is currently operational for several apps (vic, vat, a/v combo),
- promotes easy and flexible code reuse, and
- provides multiple levels of abstraction for developer.

Common framework $\Rightarrow$ leverage $\Rightarrow$ focused research!