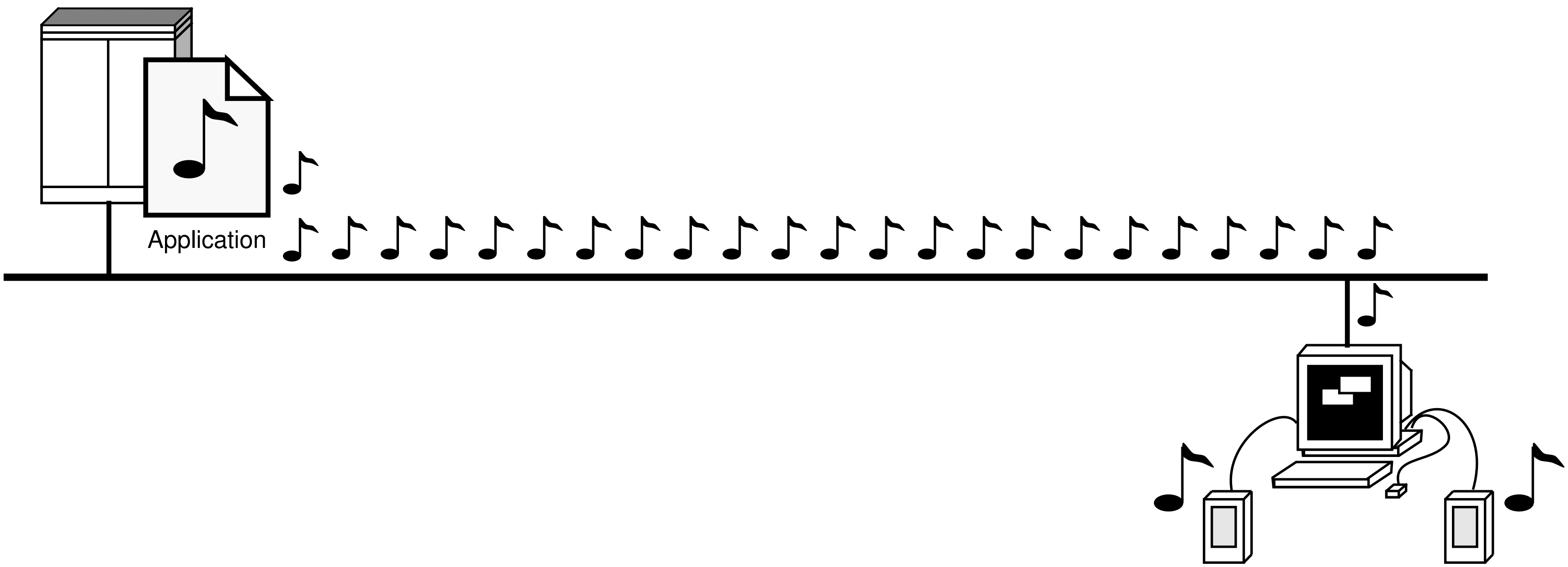
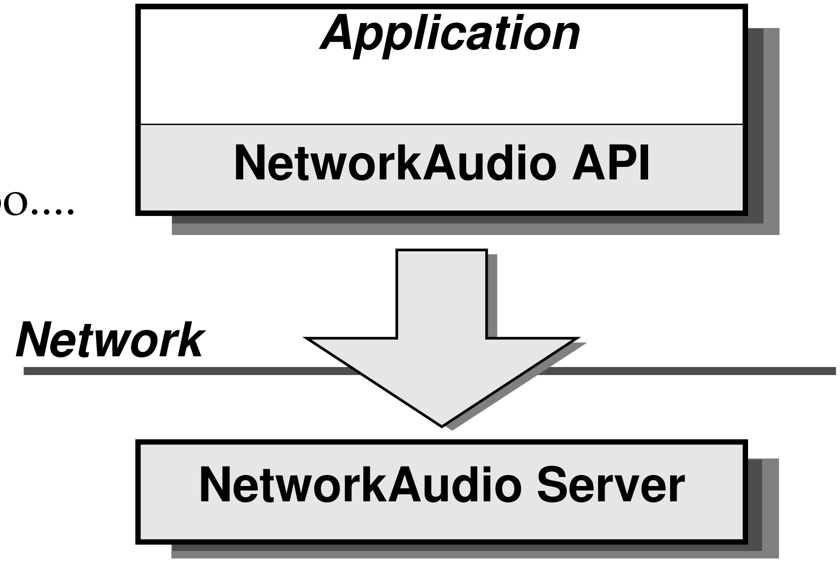
# The Network Audio System




***Jim Fulton***
***Greg Renda***
***Network Computing Devices, Inc.***

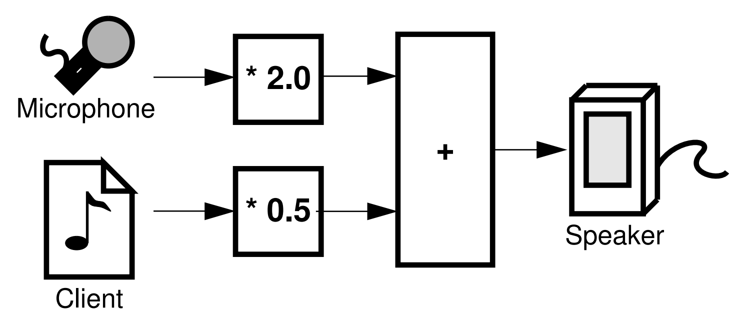
# Why Audio Over The Network?

- Applications need more than just graphics
  - Audio hardware available on all desktop platforms
  - Competition provides it...
- Desperately need a common API
  - Porting to each host unacceptable
- Choice of application host
  - X Window System model
  - Useful with Windows and Mac too....



# Goals

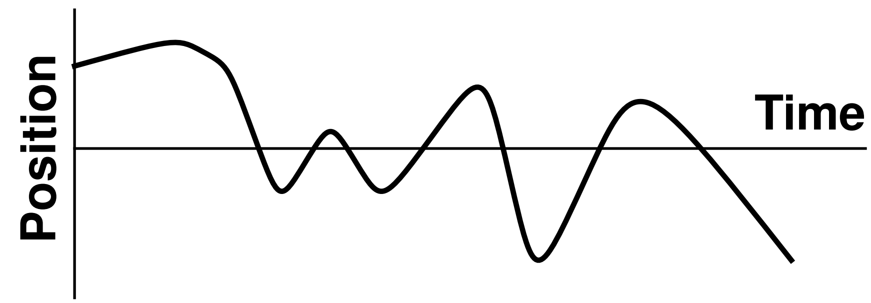
- Provide network access to audio I/O hardware
  - Network-transparent like X
  - Multiple applications at once
- Ease of Use – automatic conversions in server
  - Data formats – audio data and byte orders
  - Sample rates – resamples if applications request different rates, or if hardware can't handle
- Allow applications to "wire up" inputs and outputs
  - Any combination or sources/sinks
  - Splitting and merging multiple tracks
  - Change volume of any track



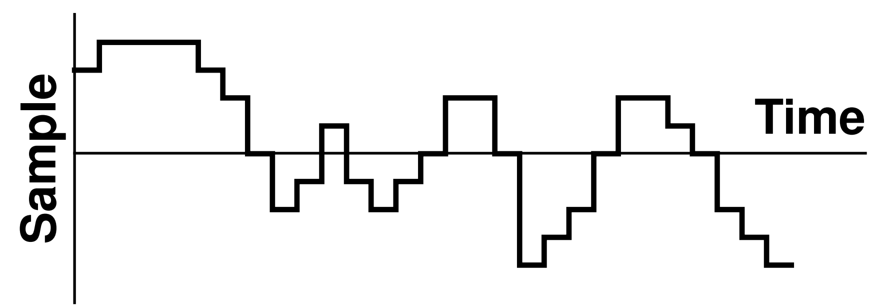
# What is Audio Data?

- Waves of compressed and expanded air; represented by position of diaphragm pushing or being pushed

- In Nature: positions are continuous

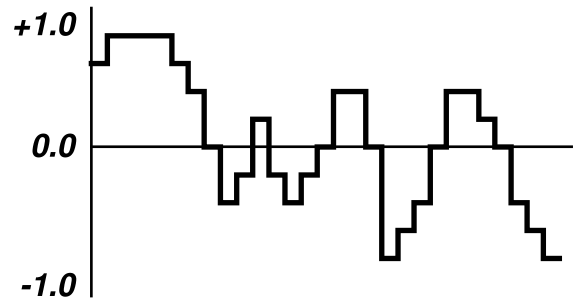



- In Software: positions sampled at discrete intervals

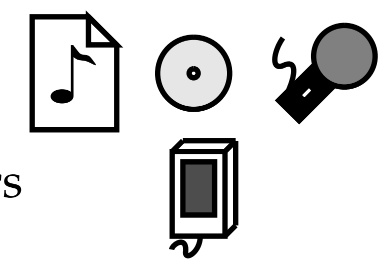


# Overview

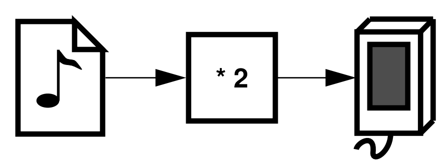
- Audio Data
  - Streams of numeric sample values [-1.0, +1.0]
  - Various encodings for values
  - Sample rate indicates granularity of stream




- Inputs and Outputs
  - Produce or consume audio data
  - Virtual and physical devices
  - Devices, network, server-stored sounds, tone generators

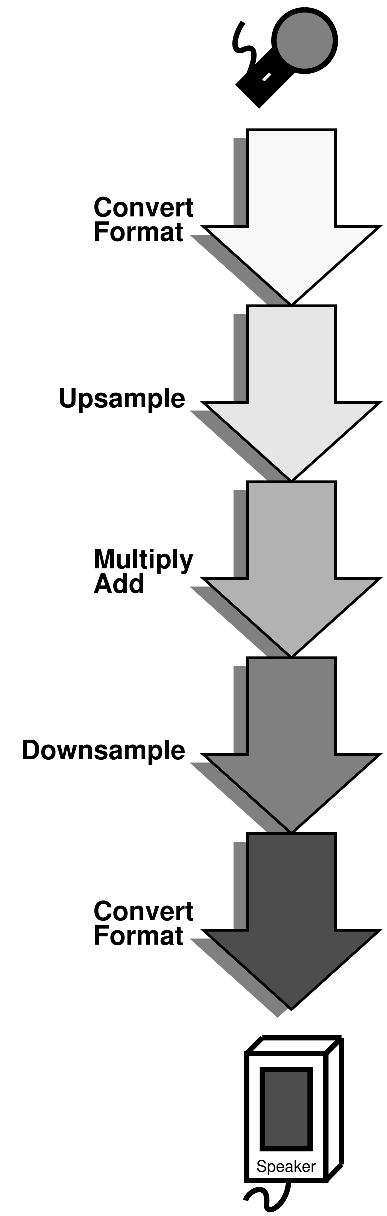


- Audio Flows
  - Inputs ⇒ Operators ⇒ Outputs
  - Operators: adders, multipliers, mixers, splitters
  - Low- and high-water marks on inputs and outputs
  - Flows can react automatically to other flows



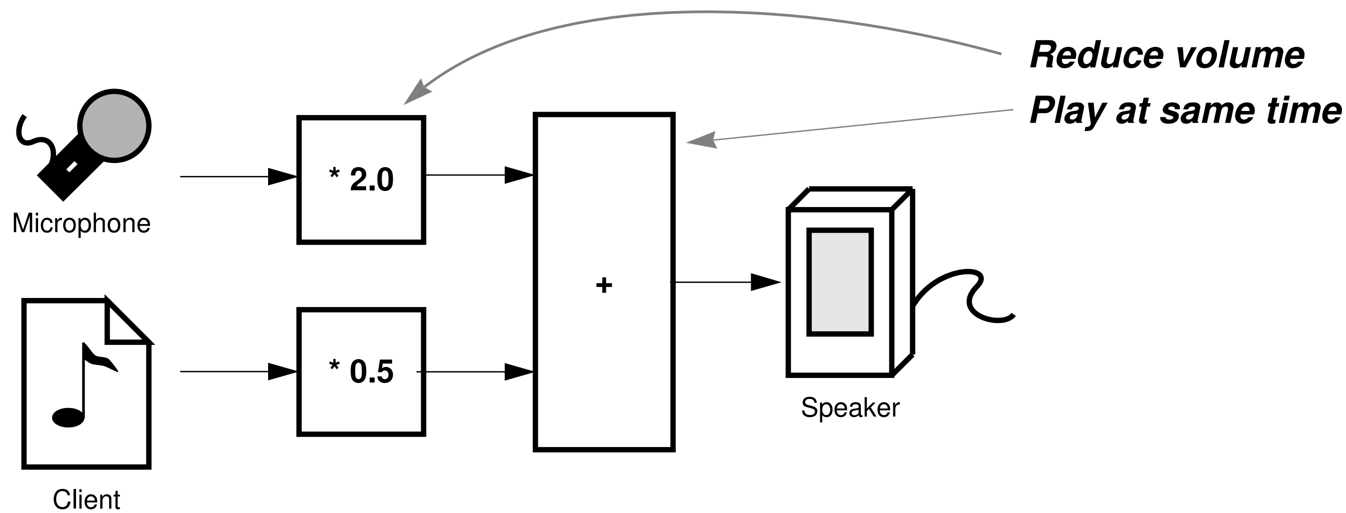
# Automatic Data Conversion

- Data Encoding Formats
  - Linear: signed/unsigned, 8/16 bit, LSB/MSB
  - µLaw (logarithmic)
  - Automatically converted into flows and destinations
- Sample Rates
  - Applications may request data at any rate from 1 kHz to 50 kHz
  - Flows run at highest of input and output rates
  - Devices asked for data at highest flow rate
  - Data automatically converted
- Output Levels
  - Data from multiple flows mixed to avoid clipping



# Flows – Software “Patch Panels”

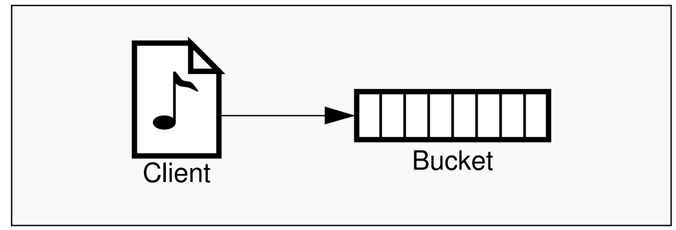
- Applications wire inputs, operators, and outputs



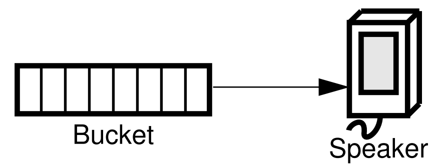

- Flows can be ***started***, ***stopped***, or ***paused***
  - Explicitly or triggered by presence/absence of data

# Examples of Flows

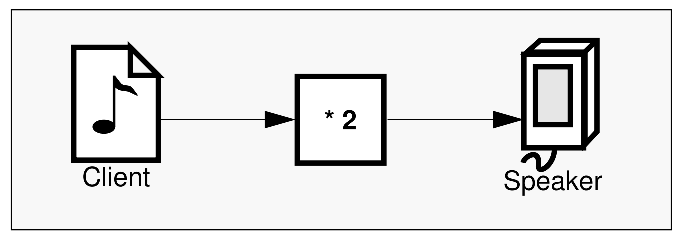
- Storing sound in Server
  - Applications can replay many times
  - Client specifies amount of data



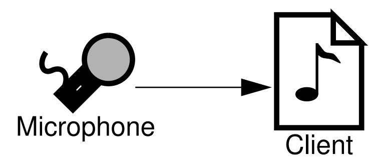

- Replaying stored sound
  - Very fast startup; no data over network
  - Applications can share buckets




- Playing directly to speaker
  - Can send infinite amount of data
  - Client sends data in chunks

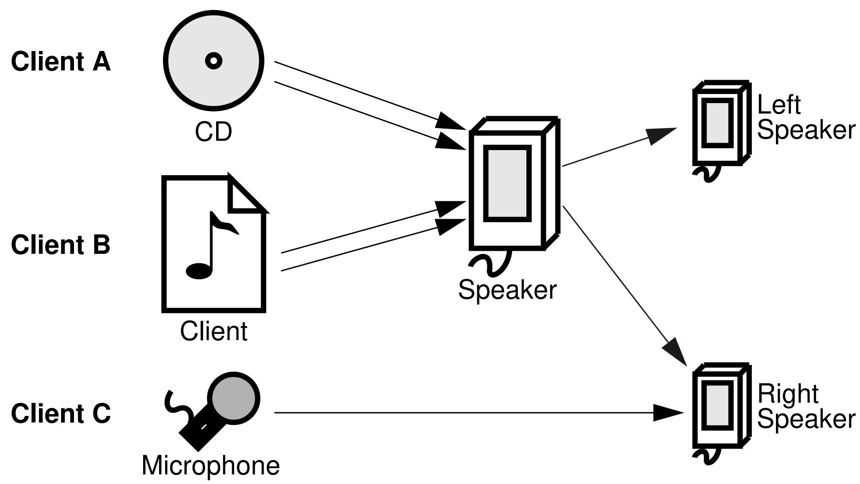



- Recording directly to Client
  - Server temporarily buffers data
  - Can fetch infinite amount of data



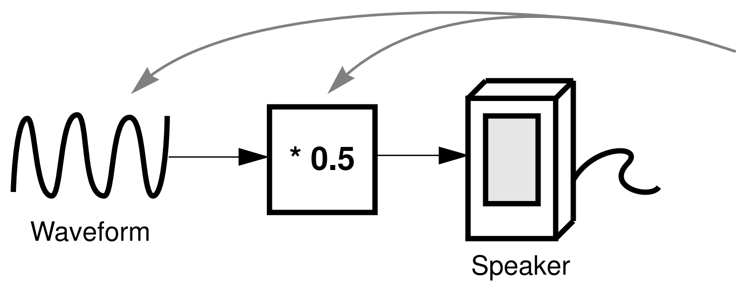
# Multiple Tracks

- Audio data can have up to 32 tracks
  - Tracks can be split and combined within a flow
- Server can provide virtual devices built from others
  - Stereo speaker could use two individual mono speakers
  - Subchannels kept synchronized (if one pauses, so does the other)



# Data Within A Flow

- Streams of values are added, multiplied, etc.
- Parameters of operators and some components can be changed dynamically
  - Allows for interactive changes in volume, tone, etc.
- Server can route data from inputs to outputs directly
  - Data only goes to clients when requested




***Dynamically changeable***

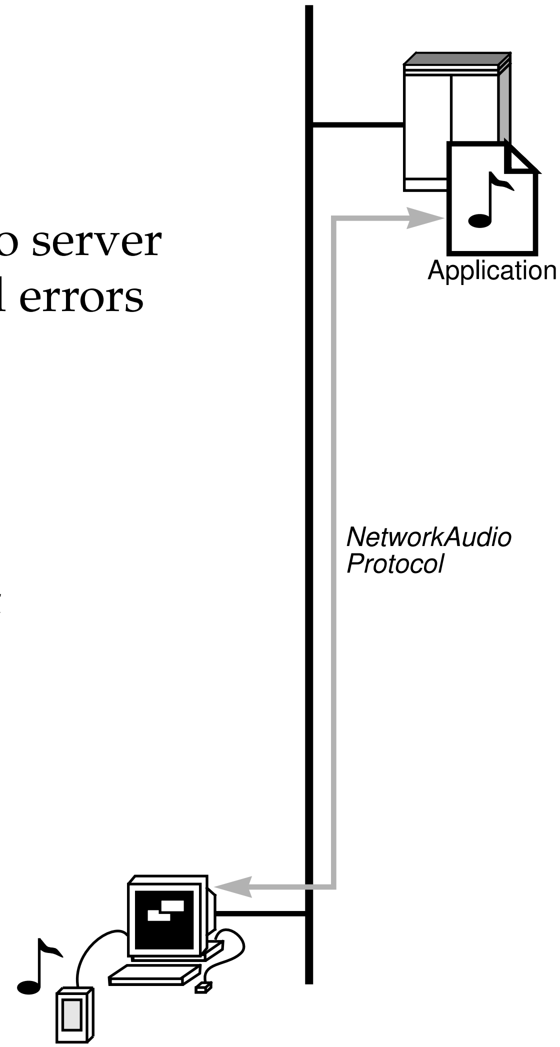
***Data does not go over network***

# Network Protocol

- Similar to X Window System
  - Asynchronous, virtual stream connection to server
  - Client sends requests, gets back replies and errors
  - Server sends events

- Multiple byte-orders
  - Client sends in native byte-order
  - Audio data of either byte-order can be sent
  - Server handles any conversions

- Miscellaneous
  - Hooks for access control
  - New versions and extensions

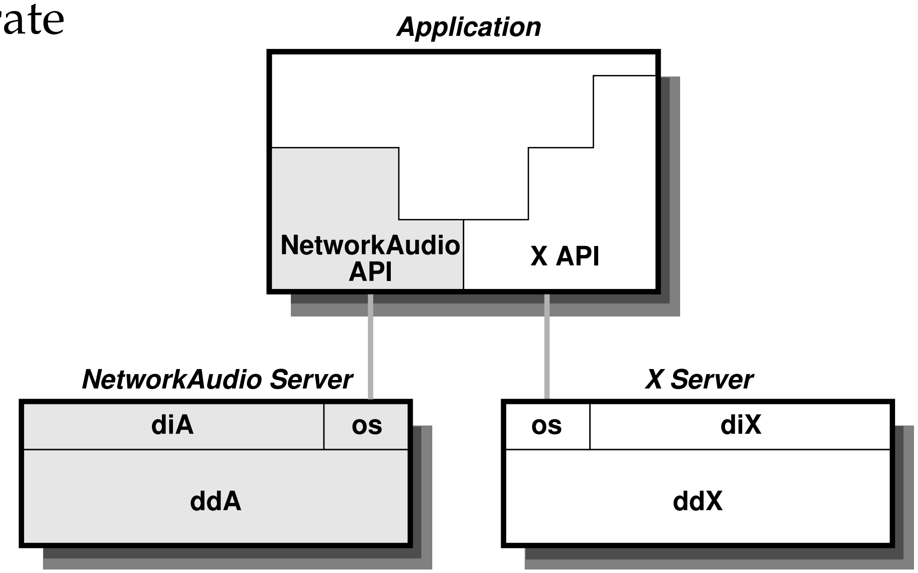


# Architecture

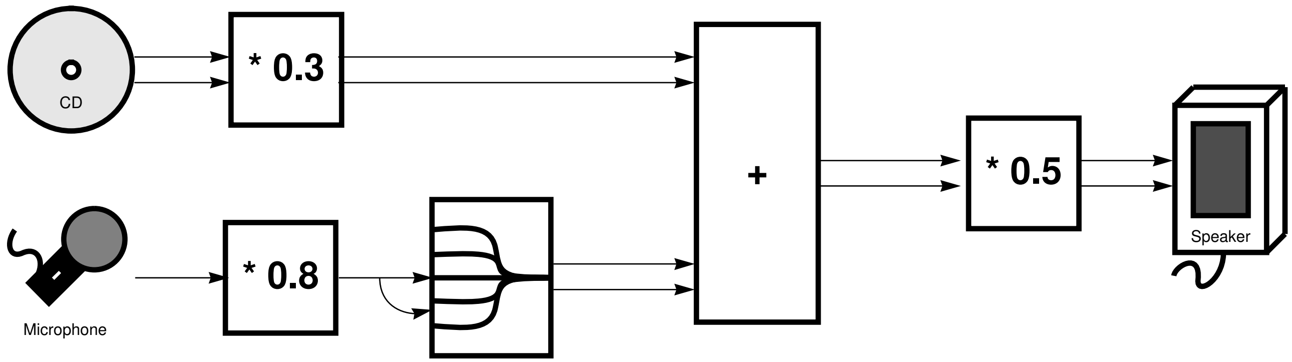
- Client/Server like X Window System
  - API is portable across application host platforms
  - Server has device-independent (*diA*) and device-dependent (*ddA*) parts

- Device-Dependent Drivers
  - Initializes devices
  - Sets gain, line mode, sample rate
  - Controls hardware interrupts
  - Reads/writes sample data

- Keep it simple, small
  - AuSun is 90k code
  - ddA is 8k code for SPARC 1 and SPARC 10



# Compiling Flows




- Each output is a function of its inputs

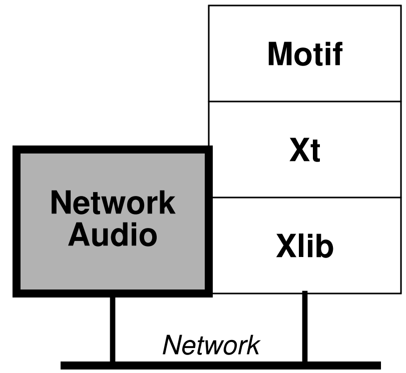
**Speaker** = ((0.3 * **CD**) + (0.8 + **MIC**)) * 0.5

- Server can compute contribution of each input

**Speaker** = (0.15 * **CD**) + (0.4 * **MIC**)

# Application Programming Interface

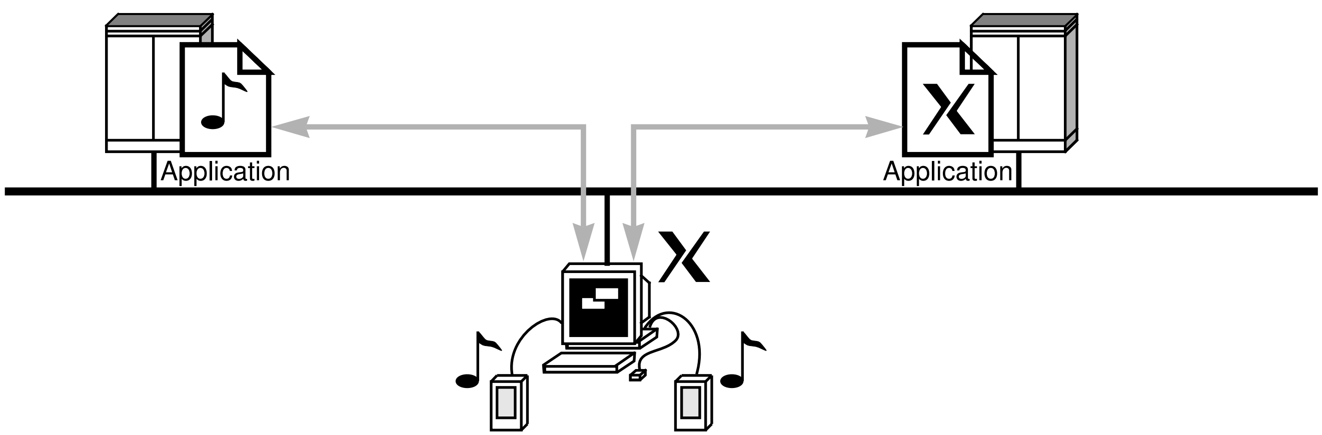
- Low-level protocol wrapper layer
  - Equivalent to Xlib
  - Provides access to protocol
  - Can be synchronous (return status) or not (get errors, like Xlib)




- Higher-level utility layer
  - Convenience routines for common operations
  - Automatic event handling
  - Single interface to playing and recording *many* file formats!
- X Toolkit interfaces
  - Just one call to register NetworkAudio connection with Xt
  - Automatically deals with events (using Xt callback)
  - Flushes audio output (using Xt workproc)

# Synchronization

- Stored sounds (buckets) can be started quickly
- Looking at integrating with X SYNC extension
  - Allow tighter synchronization with graphics
- To add time-based synchronization from AudioFile

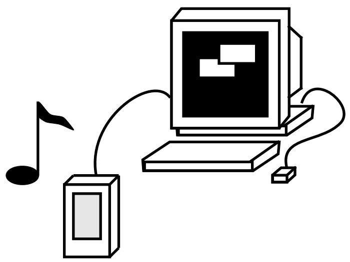


# Future Directions

- Synchronization
- More virtual devices
  - Radios for “broadcast” audio
  - Intercoms for “talk” applications
- Better device-specific control
  - How to select internal vs. external speaker?
- More radical concepts
  - Programmable filter elements in flows
  - MIDI support
  - Documentation :-)
- Work with AudioFile folks on common audio standard

# Summary

- Distributed audio in heterogeneous environments
- Portable, hardware-independent audio
  - Free server available for Sun and SGI
  - Product versions available for NCD X terminals and PC-Xware, and SCO
  - Bug your platform vendor to support!
- Support for many different audio file and data formats
- Used by a growing number of ISVs



# Where To Get Source Code

- Unrestrictive X Consortium-style copyright
  - Free to build it into products!
- Sample server for Sun, SGI
- API compiles on many host platforms
- Sample clients
  - Audio file "chooser"
  - Audio file editor
  - Audio file format converter
  - Telephone dialer and recognizer
  - *Audiotool* work-alike for use with Open Windows Deskset
- ftp.x.org:/contrib/audio/nas/

**FREE!**