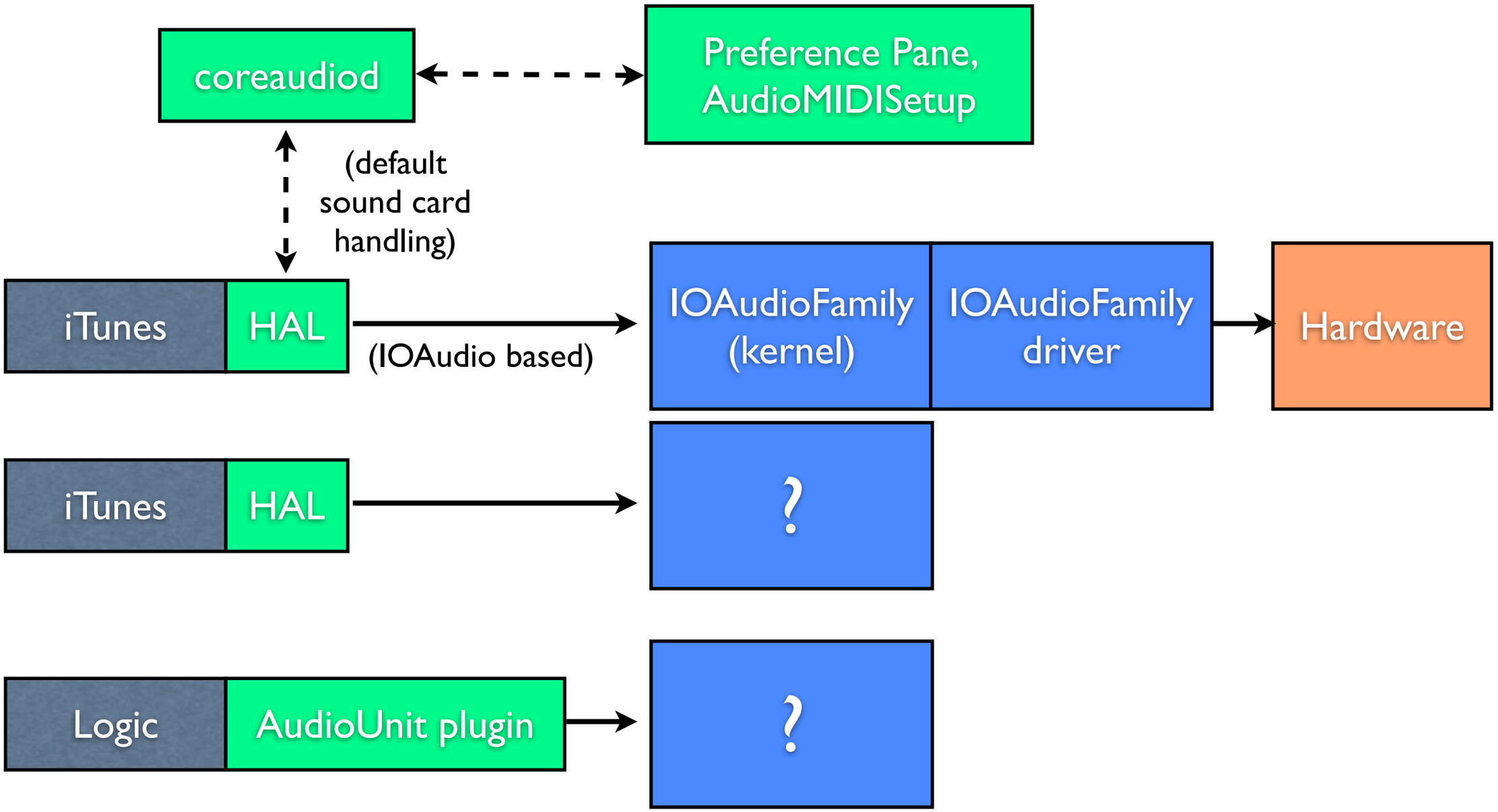
PulseAudio on Mac OS X

Daniel Mack for LAC 2011, Maynooth

Why?

- Network transparency
- Application mixing
- Free your Audio
- Interoperability with arbitrary protocols
- More testing of the PulseAudio code
- Yet another way of interacting with PulseAudio server instances in your network
- Use sound hardware that is unsupported by Linux natively
- Provide a software base for other projects (Jack for instance)

CoreAudio Overview



Darwin Kernel

- In charge to create IOAudio based sound devices which connect to real hardware
- An IOAudioDevice object represents one hardware unity
- An IOAudioEngine object represents a set of synchronously running streams

HAL

- Is loaded in the process space of each application which uses CoreAudio
- Interfaces to IOAudio based kernel implementations
- Loads user space plugins (AudioHardwarePlugin)
- Loads driver plugins (for IOAudio based drivers)

coreaudiod / System Sound Server

- Manages default sound card handling
- Plays system alert sounds
- Is always running as background daemon

Possible audio hooks

- For some application, the only way to do this is to change the default sound card (i.e., iTunes)

- Re-direct application to alternative sound interfaces
- Make the application use a special AudioUnit plugin (not always possible)
- Library pre-loading (not implemented as part of PulseAudio for Mac OS X, hackish, unsupported)

PulseAudio daemon

- poll() is broken on Mac OS X since 10.3, so we need to enable the "poll() via select() hack"
- pthread semaphores are unsupported, Apple uses an own API
- clock and time functions had to be re-implemented
- Mac OS X has a sophisticated real-time scheduling API which pulseaudiod now uses for its high-priority threads

OS specific module:
`module-coreaudio-detect`

- Scans for CoreAudio hardware which is available at load time of the module
- Registers a listener for device connection and disconnection events
- Loads/unloads modules of type `module-coreaudio-device`

OS specific module:
`module-coreaudio-device`

- Is loaded with an argument specifying the CoreAudio's internal object id
- Queries the device for its properties
- Acts as bridge between PulseAudio and CoreAudio
- Offers as many sources/sinks to PulseAudio as the actual hardware announces

OS specific module:
`module-bonjour-publish`

- Uses the native Mac OS X API for publishing network services instead of the existing module, because Avahi is not very portable due to its heavy dependency chain (DBus etc ...)

PulseAudio.framework

- A Framework bundle to abstract all PulseAudio types
- Purely written in ObjC
- Uses a clean class model, delegates and both local and remote notifications
- Is also the home for all PulseAudio binaries (such as the daemon, modules, etc)
- Has objects for service discovery and PulseAudioHelper PulseAudio server connections

PulseAudioHelper

- Runs in the background
- Reads in the Preferences plist
- Communicates with PAHALPlugin instances and the PAPreferencePane
- Forwards messages between the PreferencePane and the HAL plugins
- Can display Growl notifications
- Based on PulseAudio.framework

Preference Pane

- Communicates with PulseAudio HAL plugin instances
- Can display current settings of each connected client
- Can be used to re-route an existing connection to another server instance
- Configures the local sound daemon
- Based on PulseAudio.framework

PulseConsole

- Cocoa base tool that allows introspection of PulseAudio server instances
- Mixer GUI to make controlling volumes easy
- Multi-document based
- Listens to announces services and offers a list of server instances to connect to

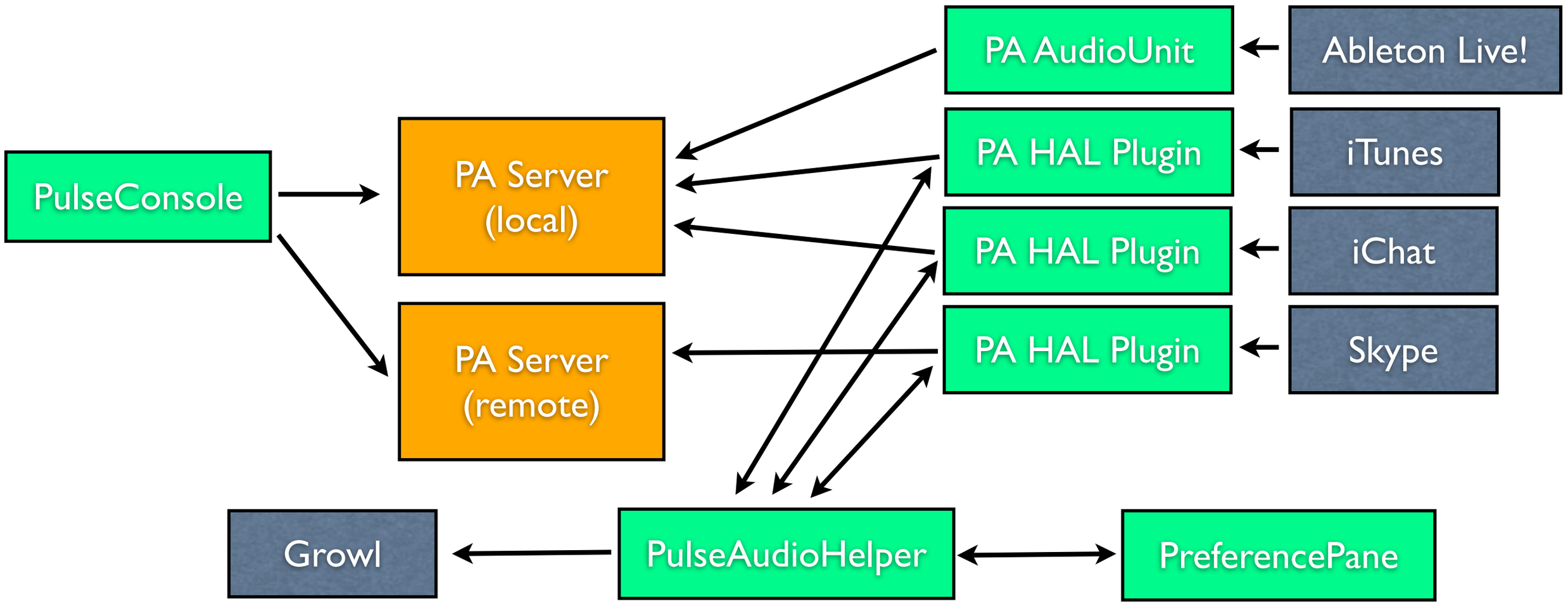
PA AudioUnit (not implemented yet)

- Will ship as AudioUnit bundle
- Can be used by all application that are capable of handling them (Apple's Logic, Ableton's Live, AULab, etc ...)
- Shall allows selection of the server to connect to
- This is not yet implemented, helpers welcome

Legacy (ignore that)

- A kernel module (kext) which registers an IOAudio device
- A user space daemon acting as counterpart
- Are not part of the binary distribution
- Exist due to historical reasons and are left in the tree as reference for other projects (you never know ...)

PulseAudio on OS X Ecosystem



Demo

(cross your fingers)

Possible scenarios

- Route Rhythmbox output to OS X machines
- Route iTunes output to your favorite Linux appliance
- Route iDVD output to Airtunes
- Use a LADSPA plugin to route sound to a machine running Mac OS X, and pipe sound through a AudioUnit host, then send it back to Linux (just an idea)
- Use PulseConsole to control the inputs and outputs of a remote machine running PulseAudio
- Use a AudioUnit plugin and send selected streams from Logic/AbletonLive!/... to a Linux host for further processing
- Interaction with uPnP/AV services in your network
- Application-based mixing of native Mac OS X clients

What's next?

- This is all work in progress and by far not yet finished
- The AudioUnit plugins need to be implemented
- The GUI components needs beautification
- Better documentation
- We need a nice website
- More testing
- Get the latency down
- More helping hands are certainly welcome
- Localization
- Bring it to the AppStore?

That's it

- Contact the PulseAudio mailing list
- All code is GPLv2
- Documentation will go to the Wiki on the project website on GitHub
- Wait for the first public release (hopefully soon)
- Binary distribution will be available

Credits

- Thanks to everyone who contributed to PulseAudio. Without a solid software base to start with, this project wouldn't have been possible.
- Thanks to Apple for providing clear and convenient interfaces to their solid audio system.