

210MTE318 - SuperCollider

an introduction

Before we start...

Course layout

— [There will be a weekly 2 hour lecture plus a 1 hour tutorial in Computer Suite 2

— [Assessment is by 3 projects worth 25%, 25% and 50%

Topics covered

— [Introduction, history, installation, architecture, using SuperCollider, the language, creating SynthDefs, creating synths, controlling synths, GUI's, synthesis techniques, routing, order of execution, ugens in detail, ordering synths, buffers, external classes and libraries, FFT's, making stand-alone applications, networking

Rules

— [All projects **MUST** be created using the Macintosh version - the other versions do not have the same functionality

— [Always comment your code - if I know what you are trying to do, then you may get a mark for it!

— [The code should be as simple to run as possible - pretend you are sending it to someone who has never used SuperCollider before. Make your documentation as clear as possible

Resources

— [You can get information from:

— The documentation that comes with SuperCollider

— The SuperCollider Users mailing list - <http://www.create.ucsb.edu/mailman/listinfo/sc-users>

— The SuperCollider Swiki - <http://swiki.hfbk-hamburg.de:8888/MusicTechnology/6>

— David Cottle has written a tutorial - email him for a copy at D.Cottle@utah.edu with the following information:

— Name/Location/website

— University/Course/Level

1) Introduction and history

What is SuperCollider?

- [It is a powerful programming environment designed for audio synthesis and processing
- [Originally developed by James McCartney
- [SuperCollider 3 is distributed under the GNU public license and is currently being worked on by a few developers (including JMcC)

A brief history...

— [SuperCollider 1 developed out of 2 programs written by James McCartney - Synth-O-Matic and the Pyrite object for Max/MSP

— [This developed into SuperCollider 2

— [Both versions were “closed-source” and “pay-for” applications which ran on Macintosh OS9

— [James then got a job (with Apple) and decided to release SuperCollider for free - however, due to licensing restrictions on some of the code used, it could not become open source

A brief history continued...

— [James then decided to start work on a new version with a radically different architecture for use on OSX

— [This version is now known as SuperCollider 3 or SuperCollider Server

— [Just to confuse things, there was a beta release for OS9 called SuperCollider 3d5.1, but this is completely unrelated to the current version 3! It was basically a continuation of version 2 with some extra image synthesis ugens. This wasn't developed any further

2) Installation

SuperCollider 3

— [SuperCollider 3 is available for OSX, Linux and Windows

— [Most of the development is focused on the OSX version - the Linux and Windows versions are ports with varying degrees of functionality

— [The Linux version is very stable, although it cannot use any of the GUI creation features (and therefore any other ugens or external libraries that require these features such as BBCut)

— [The Windows version is in very early development - only the Server has been ported, slang does not work

Installing SuperCollider (OSX)

— [SuperCollider 3 is available as a pre-compiled binary or source code

— [The binary can be downloaded from the SourceForge project site - <http://sourceforge.net/projects/supercollider>

— [The source can be downloaded from the SourceForge CVS repository

Using CVS

— [Requirements: OS 10.3 or later, libsndfile, XCode

— [Before you start, you can download libsndfile from
<http://www.mega-nerd.com/libsndfile>

— [Untar the package and compile just like any other unix/linux package:

— `tar zxvzf libsndfile-xxxx.tar.gz`

— `cd libsndfile-xxxx`

— `./compile`

— `make`

— `sudo make install`

Using CVS continued...

— [Now you can download the source code and compile it:

— [`cd /Applications`

— [`cvs -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/
supercollider login`

— [(you will be asked for a password - just hit return)

— [`cvs -z3 -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/
supercollider co SuperCollider3`

Compiling

— [When CVS has finished downloading all the sources, you can start the compile process. There is a script that does everything for you in the SuperCollider folder:

— `./compile-xcode.sh`

— [Alternatively you can build it yourself in XCode. There are 3 XCode projects to build

— [They must be built in the following order:

— `xSC3synth.pbproj -> xSC3plugins.pbproj -> xSC3lang.pbproj`

Final steps

— [SuperCollider is now ready to use - the application will be inside the “build” folder

— [You may now install any extra libraries or plugins you want to use

— [If you have multiple logins on your machine and you want everyone to be able to use SuperCollider, then make sure you change the permissions on the SCClassLibrary folder and the Synthdefs folder

Installing SuperCollider (linux)

— [There is no binary distribution - you will have to compile from CVS

— [Requirements: ALSA, Jack, libsndfile \geq 1.0, gcc \geq 3.0, pkgconfig \geq 0.14.0, autoconf \geq 2.52, automake \geq 1.4-p4, libtool \geq 1.4.2a

— [The easiest way to make sure you meet all of these requirements is to install a recent distribution!

— [I would also recommend installing the Planet CCRMA kernel and applications

Installing on a Planet CCRMA machine

If you have installed the Planet CCRMA kernel and set of packages, then you can install Jack (if you haven't already done so) using:

```
apt-get install jack-audio-connection-kit
```

```
apt-get install jack-audio-connection-kit-devel
```

You can also install QJackCtl to configure and control Jack:

```
apt-get install qjackctl
```

Install libsndfile using:

```
apt-get install libsndfile
```

```
apt-get install libsndfile-devel
```

Linux installation continued...

— [download the source from CVS as before. Compile using:

— `./linux/bootstrap`

— `./configure`

— `make`

— `sudo make install`

— [The server is now ready to use. However, there are a couple of steps left if you want to use emacs to interface with slang (recommended!)

Setting up emacs

[In the linux/examples folder there are two files called `sclang.cfg` and `sclang.sc` - copy these to your home directory and rename them so there is a `.` at the start of the name

— `cp sclang.cfg /home/username/.sclang.cfg`

— `cp sclang.sc /home/username/.sclang.sc`

[Edit the `.sclang.sc` file in your home directory for your machine - in particular you may need to change the `Server.program` setting and the number of inputs and outputs on your audio card

An example .sclang.sc

```
— [// set up Server to use your executable
— [Server.program = "/usr/local/bin/scsynth";

— [// same for Score
— [Score.program = Server.program;

— [// set some server options for a different setup
— [#[\internal, \local].do { |s|
—   s = Server.perform(s);
—   s.options.numInputBusChannels = 8;
—   s.options.numOutputBusChannels = 8;
— };

— [// hook up jack ports to audio channels
— ["SC_JACK_DEFAULT_INPUTS".setenv(
—   "alsa_pcm:capture_1,"
—   "alsa_pcm:capture_2,"
—   "alsa_pcm:capture_3,"
—   "alsa_pcm:capture_4,"
—   "alsa_pcm:capture_5,"
—   "alsa_pcm:capture_6,"
—   "alsa_pcm:capture_7,"
—   "alsa_pcm:capture_8"
— );
— ["SC_JACK_DEFAULT_OUTPUTS".setenv(
—   "alsa_pcm:playback_1,"
—   "alsa_pcm:playback_2,"
—   "alsa_pcm:playback_3,"
—   "alsa_pcm:playback_4,"
—   "alsa_pcm:playback_5,"
—   "alsa_pcm:playback_6,"
—   "alsa_pcm:playback_7,"
—   "alsa_pcm:playback_8"
— );

— [// EOF
```

Setting up emacs continued...

— [Now edit your emacs configuration file (.emacs) by adding the following 2 lines:

— [`(add-to-list 'load-path "/path/to/SuperCollider3/linux/scel/el")`

— [`(require 'sclang)`

— [SuperCollider is now ready. Start Jack, then start emacs with the option -sclang:

— [`emacs -sclang mytestfile.sc &`

Installing SuperCollider (Windows)

I would recommend using the linux version if you are stuck with a PC. The Windows version has a lot of bugs and has very limited functionality. This will not be accepted as an excuse for incomplete projects.

If you really want to try it, there is a package available on the SourceForge project page (same url as before). Instructions should be included. You will need to install Python.

3) Architecture

scsynth & slang

— [SuperCollider 3 works on a client-server basis

— [scsynth is the server

— [slang is the client used to send data to the server

— [Other clients which support the OSC protocol may be used -
for example Max/MSP, PD, Java, Flash, Director

The server

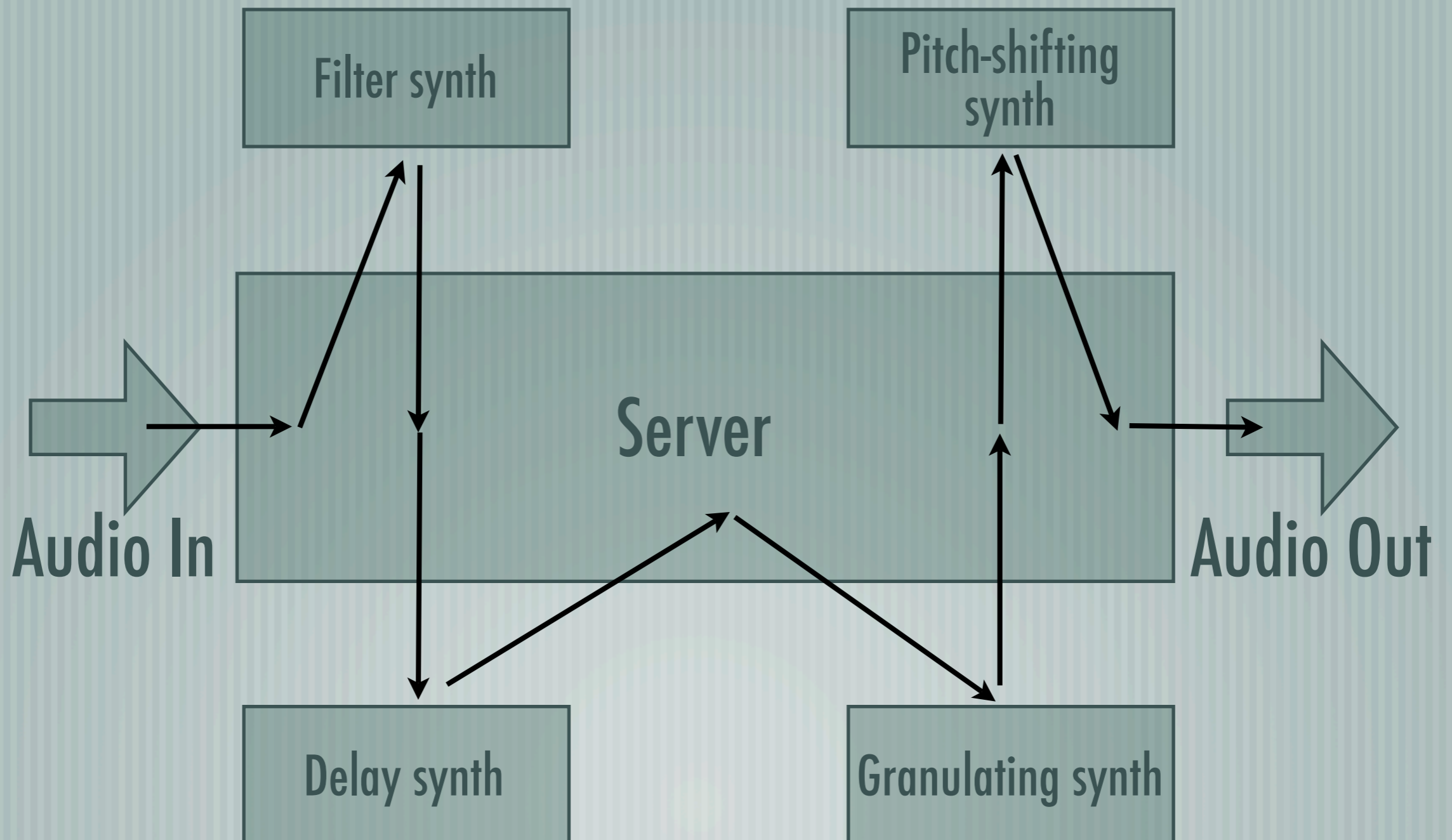
— [The server is what makes SuperCollider so powerful

— [You create multiple synths on the server - each synth becomes a "node" and information (audio and control) can be shared between them

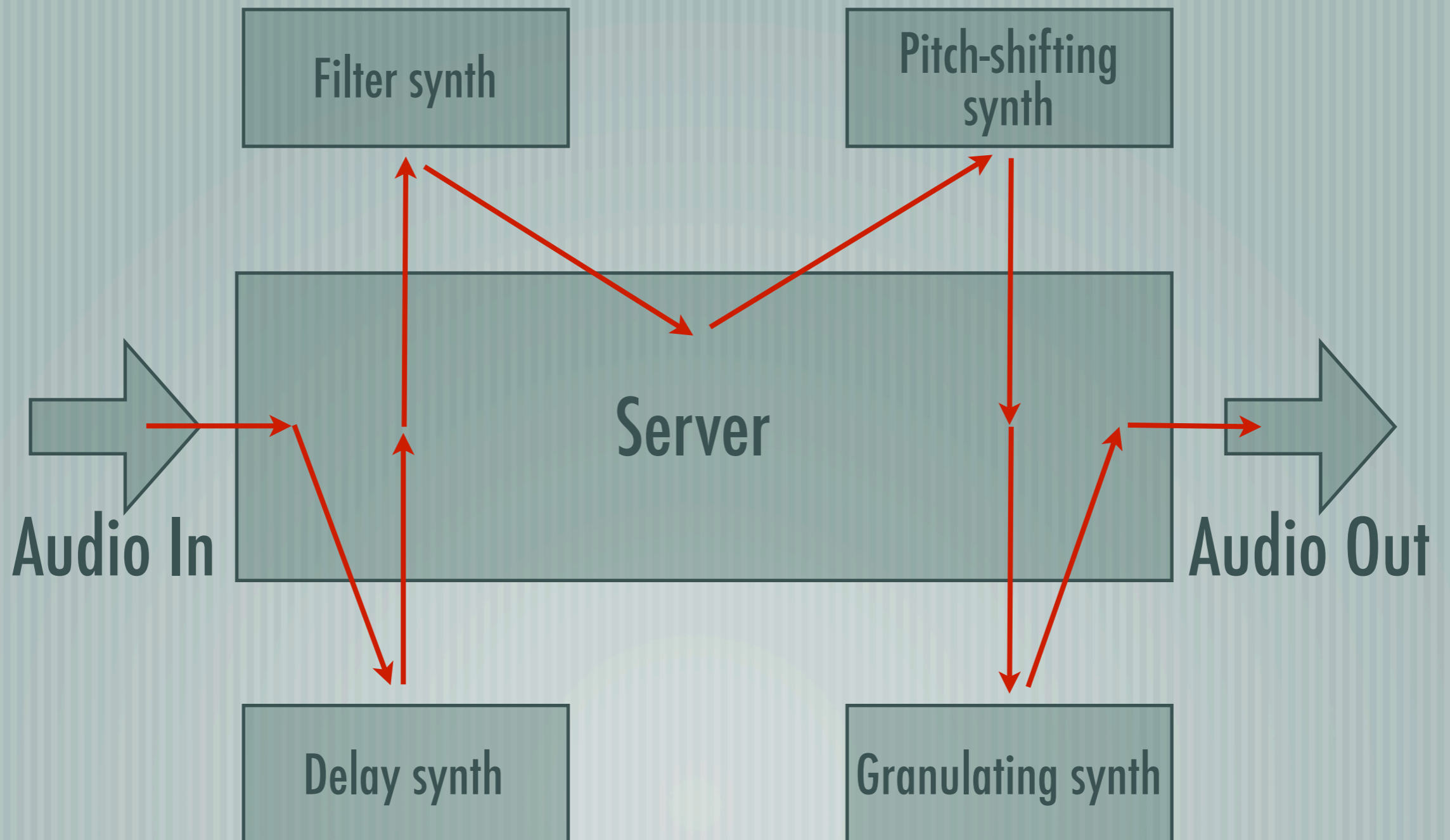
— [nodes can then be grouped together as "Groups" (funnily enough) and be controlled at the same time

— [You can also have servers on different machines and send data over TCP or UDP

The server continued...



The server continued...



Busses

— [Synths write audio and control data to a bus on the server

— [Other synths can then receive this data from the bus and write their output to the same or another bus

— [Each bus is assigned an individual ID. The audio inputs and outputs of your sound card are always the first few busses on the server

sclang

— [The client is used to define synths and pass them on to the server. It is also used to generate GUI's and send control information from the GUI and other input devices to the server

— [Synths are defined by "SynthDefs"

— [These SynthDefs will contain information about how the synth operates - where it gets audio/control information from, which bus/busses it writes to.

Ugens

— [The SynthDefs will also contain references to a number of Ugens (unit generators)

— [These Ugens are objects that process or generate audio

— [They can normally run at 2 rates - audio rate (.ar) and control rate (.kr)

— [Audio rate ugens produce samples at the sample-rate the server is running at

— [Control rate ugens produce one sample per control cycle - this is 64 audio cycles by default, but may be changed by the user

4) Using SuperCollider

Start-up

- [Whenever you start the program, you will be presented with 3 windows: a “post” window which SuperCollider uses to post messages and the two default server control GUI’s
- [The localhost server runs as a separate application to the client
- [The internal server runs within the same space as the client and can access shared memory

localhost vs. internal

— [Because the internal server shares the same memory space as the client, messaging latency is reduced. You can also use scopes to display the contents of busses

— [However if the client crashes, so does the server

— [If you use the localhost server, messaging latency increases slightly. However if the client crashes, the server keeps producing audio

— [The default server is set to be the localhost server. It is also defined as the interpreter variable "s"

Using the interpreter

— [When you open an existing file or create a new one, it is opened inside an interpreter - there are a number of shortcuts which you can use to control it

— [To send a line of code to the server, put the cursor at the start of the line and press enter. Note that this is not the same as return - it is the enter key on the numberpad you want!

— [To send multiple lines, highlight them all and press enter. As a shortcut you can surround your code in parentheses and double click one of them - everything in between will be selected

more shortcuts

— [To stop the server processing your code, press apple-period

— [You can comment out a line of code by putting `//` at the start of it

— [You can comment out multiple lines of code as well - start the comment with `/*` and end it with `*/`

— [You can also syntax colourize your code to make it easy to work with - select it and press apple-'

— [To view a help file for a ugen, double-click on it and press shift-apple-?

— [To look at the class-definition of a ugen, select it and press apple-J

210MTE318 - SuperCollider

an introduction