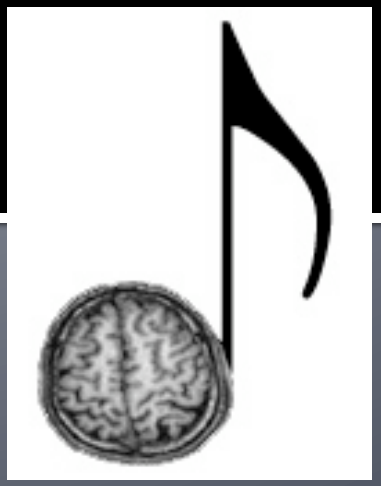
How changeable music changes against the composer's will; SuperCollider development, interactive and generative music programs



Nick Collins, University of Sussex

Themes

- The context of maintenance
- Challenges in SuperCollider development
- Specific cases studies from SuperCollider users and my own experience

Why change?

- Inexorable commercial pressure (competition, funding per release)
- Inexorable artistic pressure (inspiration, curiosity, and baser motivations)

Though a moratorium might be nice…

Miguel Depedro (Kid 606) noted:

‘By the time you've learned how to use something, there's already something else. I would love for everything to just pause right now - no new advances, no faster computers, no new Max. And then we'll see what we do for the next two years.’

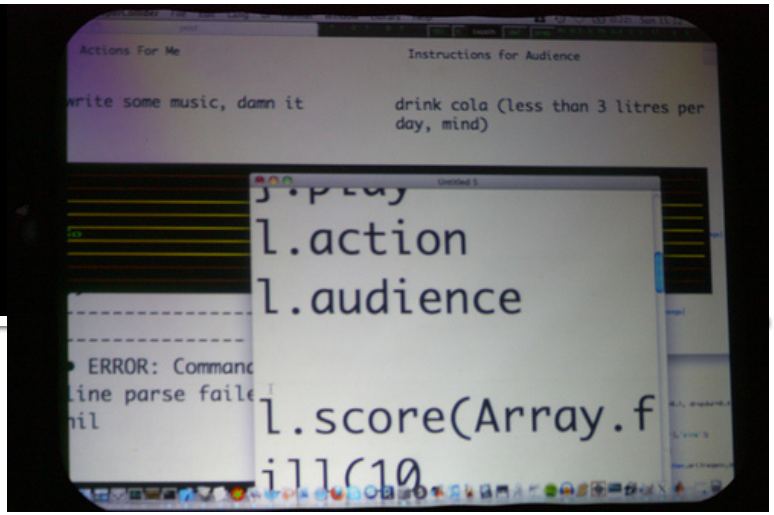
(2002, Songs in the Key of F12: http://www.wired.com/wired/archive/10.05/laptop.html)

The improviser's solution

- Don't try to preserve anything
- Let things be of their moment
- Irrevocable change is good, design cycles are not realtime constructions

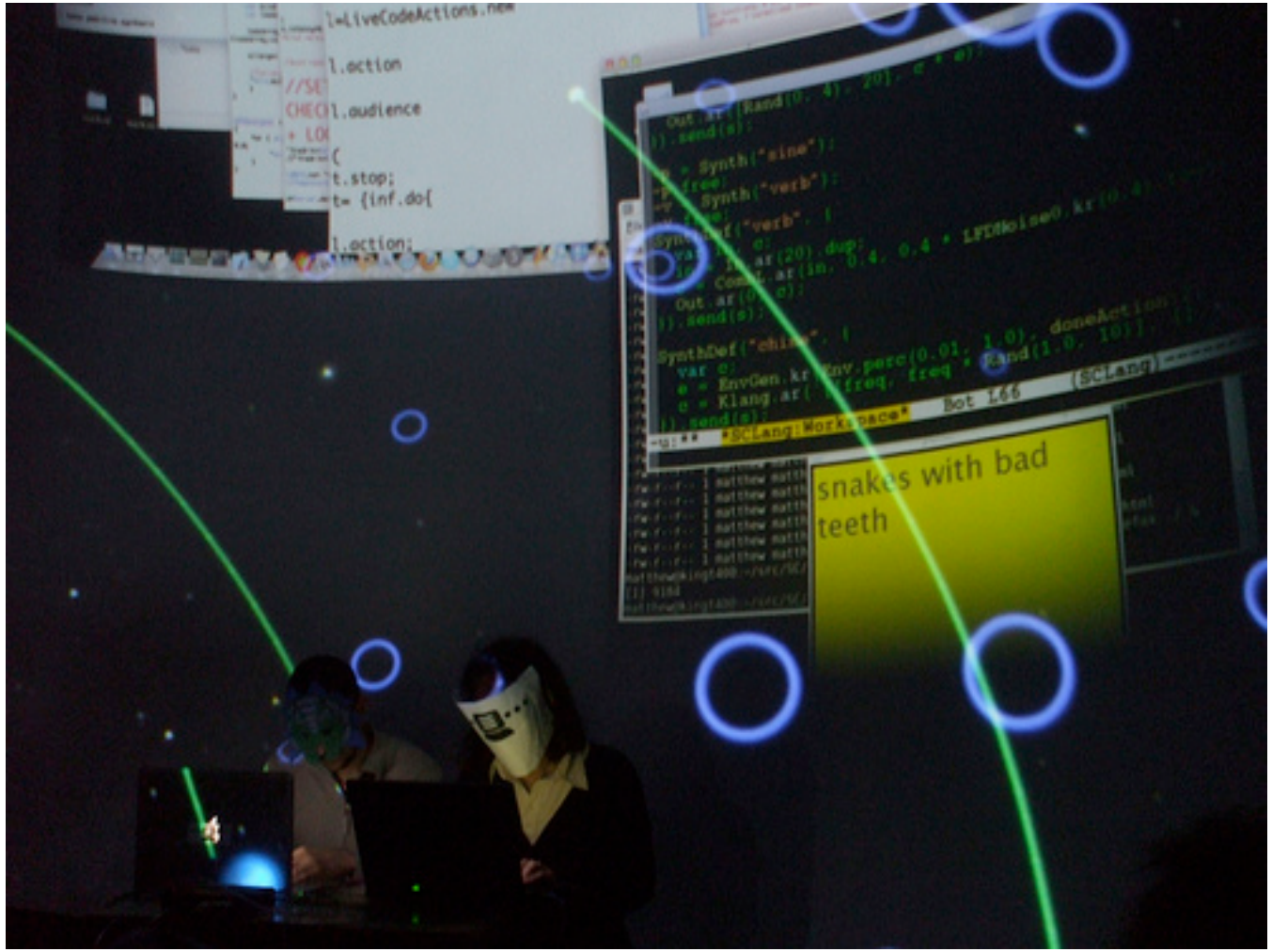
Improv Example:

Live coding



- Usually a bit controversial
- Art of re-programming during a concert; changing your mind about a process once established
- Increasing aspects of performance art
- Venues including studios, concert halls, planetariums…

Wrongheaded



Warning!

- Look away if you're squeamish before the next slide

An anatomy theatre

There is a serious side

- Algorithmic choreography
- Doesn't have to involve computers
- Rewriting the rule book of rule-based art?
- Insights into the psychology of programming, and musical activity
- Conceptual level preservation

But…

- Assuming we have invested a lot of time in developing systems, what is the best survival strategy?

The sci-fi bit; far future survival

- Work your piece into the genetic code of the cockroach (F. Olofsson)
- Create an institute dedicated to preserve your work or culture in general (Jem Finer's Longplayer, Long Now Foundation)
- Create works which are potentially infinite but can be appreciated by sampling or analysis
- Etc, see also my article "Infinite Length Pieces: A User's Guide"

A more practical study: SuperCollider

- Three major versions, all relatively incompatible with each other: SC 1 (1996), SC2 (1998), SC3 (2002-now)
- Open source, small dedicated development team
- Contributions can be ad hoc based on availability; 'do-ocracy'

Maintenance Issues

- Supporting Mac, Linux, Windows
- Historically a Mac program, now fully cross-platform; but less Windows devs.
- Some platform specific code, two GUI systems, one Mac specific (Cocoa) and one based on Java (SwingOSC)
- Chasing operating system updates; one current issue is supporting OS X 10.4 to 10.6; fat universal binaries for 32 and 64 bit, some PPC machines rather than Intel still around

SC Changes

- SC 2 to SC3 transition meant a lot of compositions had to be abandoned, or converted at great expense of time
- PPC to Intel switch on Mac caused subtle changes (PPC dealt with divide by zero much better!)
- Development tug of war between backwards compatibility and rationalisation/efficiency/ new features

Examples

- Updated to 3.4 last weekend
- Deprecation of memStore:

ERROR: Method 'memStore' of class SynthDef is deprecated and will be removed. Use 'SynthDef:add' instead.

- Change in plug-in Unit base class: would crash as soon as run any old plug-in, must re-compile everything, and two versions of SC on one machine which share plug-ins can't co-exist

Dev to-and-fro; a 'solution'

```
struct SC_Unit_Extensions {float * todo; };

struct Unit {
struct World *mWorld;
struct UnitDef *mUnitDef;
struct Graph *mParent;
uint16 mNumInputs, mNumOutputs;
int16 mCalcRate;
int16 mSpecialIndex; //used by unary and binary ops
int16 mParentIndex;
int16 mDone;
struct Wire **mInput, **mOutput;
struct Rate *mRate;
SC_Unit_Extensions* mExtensions; //future proofing and backwards compatibility; used to be
                                 //SC_Dimension struct pointer
float **mInBuf, **mOutBuf;
UnitCalcFunc mCalcFunc;
int mBufLength;
};
```

Fredrik Olofsson's example

PPC vs Intel, or an Internal soundcard driver issue…

{RHPF.ar(GbmanN.ar([2300,1150]),LFSaw.ar(Pulse.ar(4,[1,2]/8,1,LFPulse.ar(1/8)/5+1))+2)}.play

Deliberately blows up high pass filter, new software limiter doesn't cope well

'Power PC'/no limiter

'Intel'/limited

http://www.fredrikolofsson.com/foblog/?q=node/394

SC-user discussion

- Informal survey on SuperCollider mailing lists
- Range of views, probably energised longer term users more; newer perhaps not experienced enough to worry
- Sanity check from Thor Magnusson: 'Let the old stuff rot, this is digital media's way of filtering out rubbish. Things are good in their time when they're relevant and then they become obsolete. Do you think future generations will be as obsessed with archiving as we are?'

Keeping track of dependencies

- It's not just a particular version of SC application and code; also, third party extensions to preserve, and the computer and OS itself! ' I guess I should be more organized and keep track of these things…' (Miguel Negrão)
- Adopting the model from Python ('virtual_env and/or buildout') would be good, (Chris Sattinger)
- SC does have a standalone facility, but OS X only

Fabrice Mogini

”It took me a long time to adjust to the language changes from OS9 to OSX and I lost many compositions written in old code. It doesn' t matter because we do change as well…we may use this opportunity to work on fresh ideas but I realized for instance that the high number of nested Spawns I was using in SC2 was giving very satisfying musical results at the time… so I guess, all the limitations we may have at any time (speed, language) are a challenge and platform for creativity and we can just do sounding great stuff at any stage…”

Josh Parmenter

- 'Porting … works from SC2 to SC3 is something I hope to never have to do again.'
- 'just document the work the best I can' Experience of producing, and documenting, Richard Karpen's *Aperture* (2006), see http://faculty.washington.edu/karpen/Aperture.html
- 'I can honestly say if I was offered the trade-off of having a sustainable model that I could work in (and know my work would survive in to the next century) but I had to give up recording technology, I would take the first in a second.'

Andrea Valle

- Autography (own hand) vs allography (another writing your words)
- Documentation via original footage, and higher level abstract descriptors; involved in reconstruction of Poème Électronique (CMJ 33(2), 2009). Notes that plans/scripts easier to work with than original materials
- High level notational systems for electronics vs exact code; contrasted Centro Tempo Reale in Italy, and IRCAM's old max patches

Marije Baalman

- 'I'm not sure all my home-made performance interfaces and analog electronic components still work, and that may be harder to fix than the code itself.'
- 'But, for the majority of these pieces, I consider them as studies in my artistic development, and as such have no wish to perform them again, as I have moved on, or developed techniques further.'
- For last five years creating collaborative dance theatre performances and installations, mixing multiple technologies; balance between adapting to specific ideas in rehearsal, and re-usable code.

Maintaining own compositions, interactive systems, and tools

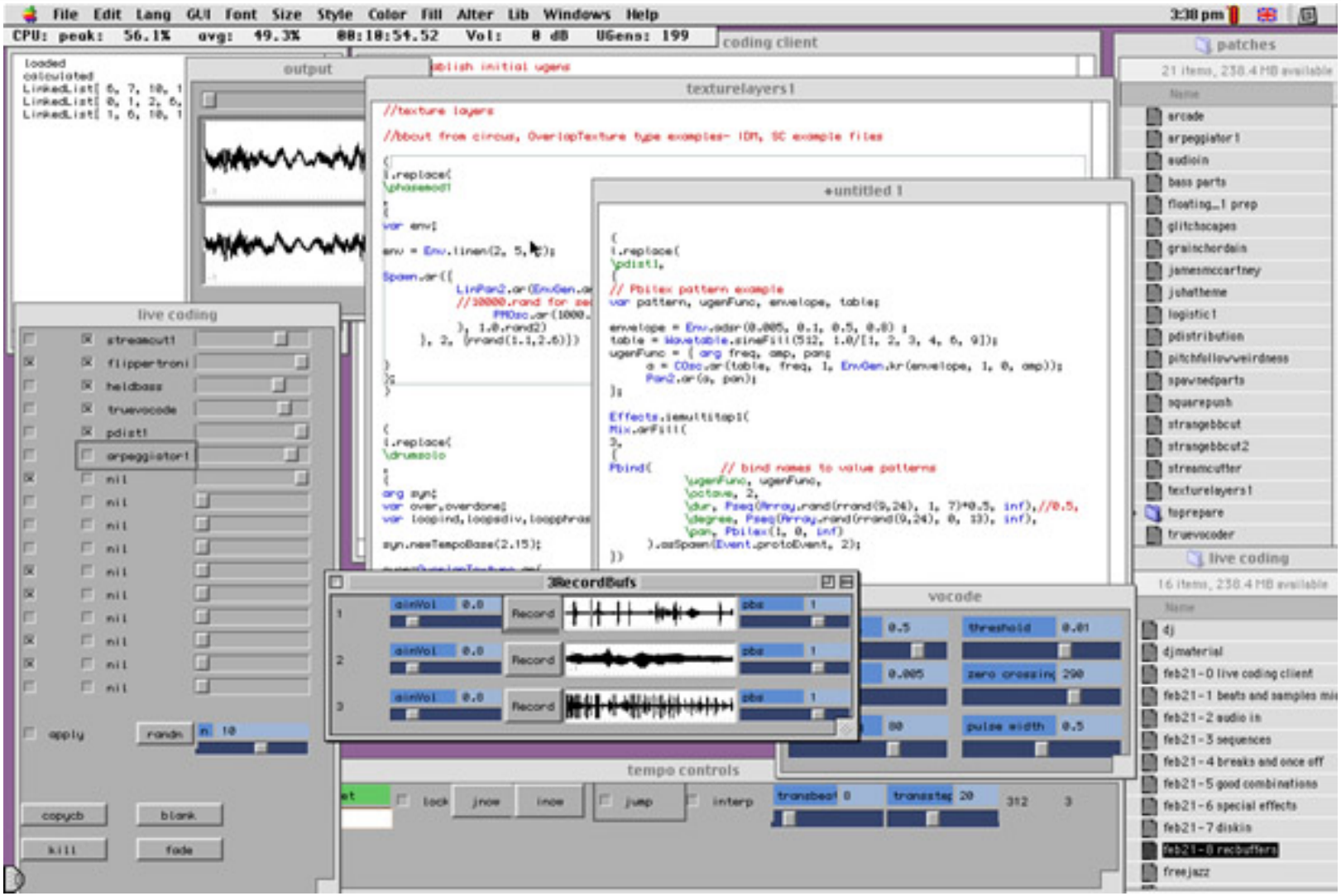
- Circa 2006 interactive music systems no longer work comfortably (takes about an evening to update one, mainly plug-in fixes/rebuilding)
- BBCut1 and 2 still work, but not supported since 2008
- Lots of sound synthesis and machine listening work in SC core or sc3-plugins project; torn between backwards support and desire to re-write!
- Duty of care too great when not actively working on further extending old systems; quandary for a researcher where pressure is to investigate new areas!

Fixing plug-ins can destroy old sounds

- Updates for safety may have unintended consequences on timbre
- Have created plug-ins in the past with theoretical errors; but released. Have to choose whether to change the sound to fix theoretical purity, or make version 2!
- Example: GravityGrid and GravityGrid2

Live Coding system

SC2 version:



SC3 version…

- If preparing on a new computer, issues with sample paths
- But still works otherwise; avoided third party libraries in the main, amazed own bbcut not broken yet…

Creating new plug-ins

- Wanting to now release for SC and Max/MSP
- Need common code for main DSP and small wrapper for each
- SC in particular has strong real-time memory allocation constraints, since everything can be dynamically created without stopping audio
- Example: LPCAnalyzer and lpcanalyzer~

Max/MSP collaboration

- Working with some other composers, releasing prototype plug-ins
- May eventually lead to general release, but exclusive use as testing group and more controllable research project

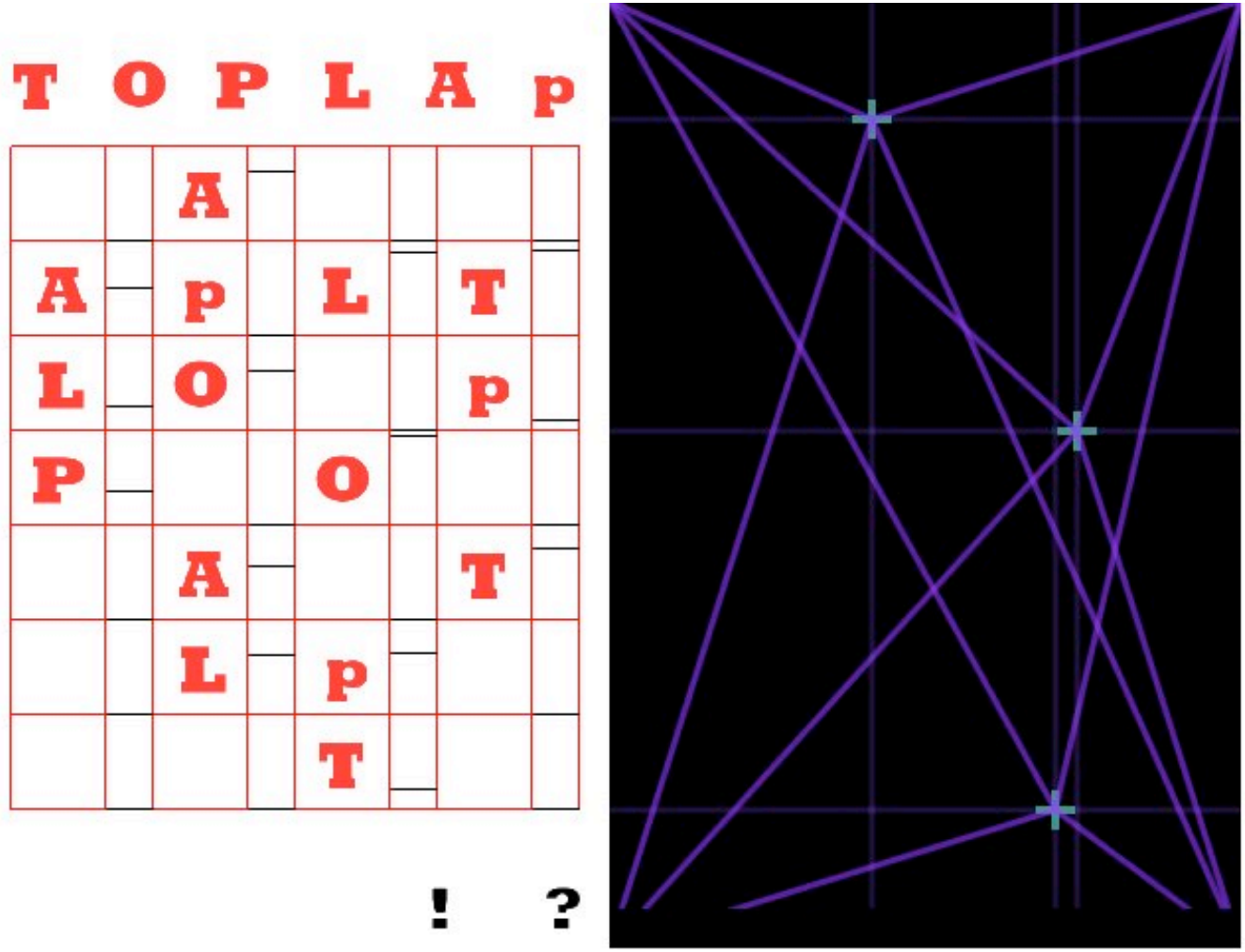
The temptation of pure C

- Writing some applications now independently of SC
- Return to pure C after intermediate years of a lot of research within SC
- Avoid licensing issues, write vanilla C++, 'Rowe's model'

iPhone

- Fun device, even if rather proprietary
- Useful source of controllers; 3 axes of accelerometers, audio input (bit more awkward on iPod Touch), up to five touches detected simultaneously
- Will 'soon' be obsolete

TOPLAPapp and iGendyn



iPhone Developer Dilemma

- Just had to renew developer license
- For the moment, do want to create a few more Apps, and keep the current ones available/update
- But certainly not mad about paying every year here on
- To go to Android?

Still…

- Perhaps preservation is not compatible with such a high speed of technological shift
- Perhaps composers should not consider themselves as too privileged, but just one amongst billions of human creators
- Won't future media archaeologists be busy with their own work? Archaeology on a Friday, or a professional cult?

Preservation

- Preservation more robust via higher level description, return to specification when re-making works
- Recognition of recordings as an essential adjunct
- If only the devil wasn't in the details

Thinkyou for lastening