soundsoftware.ac.uk

SoundSoftware.ac.uk: Towards Reusable Software for Audio & Music Research

Mark Plumbley, Chris Cannam and Luis Figueira
Centre for Digital Music
Queen Mary, University of London

EPSRC
Engineering and Physical Sciences Research Council

Overview

- Reproducible research
- Survey
- Research Pipeline problems
- Barriers and approaches
- Where do I start?
- Suggestions for research groups
- Conclusions

Reproducible Research

(Buckheit & Donoho, 1995; Vandewalle et al, 2009)

Idea: researchers should be able to reproduce the work of others.

Research used to be "reproducible" from the paper alone.

In audio & music research, methods are now too complex.

The paper is not enough: need algorithm, parameters, datasets, ...

So, we need

- The paper (ideally Open Access)
- The code (ideally Open Source)
- The data (ideally Open Data)

Well-known example: WaveLab (Buckheit & Donoho, 1995)

But in audio & music research, few people do this. Why?

Research software in practice

We carried out a **Survey of UK audio and music researchers***.

80% of respondents reported developing software, but

only 40% of those said they took steps to reproducibility

only 35% of *those* reported ever publishing any code

i.e. only **15%** tried to be reproducible and published the code.

Also: 51% said their code never left their own computer

Even published software can be hard to reuse, due to the variety of technologies in use: MATLAB, Max/MSP, C++, HTK, MPTK, SuperCollider, Python, Scheme, Prolog, Clojure...

* - Oct 2010-Apr 2011, 54 complete + 23 partial responses.

Why don't people publish code?

We found:

- Lack of time
- Copyright restrictions
- Potential for future commercial use

Other factors (UK Research Information Network, 2010):

- Lack of evidence of benefits
- Culture of independence or competition
- Quality concerns (self-taught programmers)

Also: it takes effort early in the research cycle;
hard to find time/motivation after the paper is published

So instead of this Research Pipeline,

Researcher A ("Producer")

- Read background papers
- Do own research
- Publish paper X

Researcher B ("Consumer-Producer")

- Read paper X
- Understand/reproduce results in paper X
- Do more research building on X
- Publish paper Y that cites X / produce product that uses X

... and so on.

... we have: *Real* Research Pipeline

Researcher A ("Producer")

- Read background papers
- Do own research (including lots of coding)
- Publish paper X (not enough space for all the code)

Researcher B ("Consumer-Producer")

- Read paper X
- Can't reproduce or use results in paper X
- Tear out hair
- Give up / do something else

NB: A and B may be in same group (or same person later!)

How can we solve this?

We're taking a **bottom-up approach**:

- Make **incremental improvements** to development practice

  by

- Identifying **specific barriers** to publication and reuse, that are relatively straightforward to address

So we hope to:

- Increase perception among researchers that code is something you can work on together, that can be reused
- Prepare the ground for reproducible publication

Barriers to publication and reuse

- Lack of education and confidence with code
- Lack of facilities and tools
- Lack of incentive for publication
- Platform incompatibilities

Barrier: Lack of confidence in code

Issue: Researchers largely self-trained in software development

Our approach: Training in research software development

Relatively small amounts of training can pay off

Autumn School (Nov 2010) based on Software Carpentry

- Version control systems
- Unit testing, test-driven development
- Python syntax and structure
- Managing experimental datasets

Further schools planned for spring—summer 2012.

Also planned: Tutorials at conferences

Barrier: Lack of facilities and tools

Issue: Researchers don't use code hosting / version control

- Research groups / institutions often do not provide any
- Researchers often unaware of them

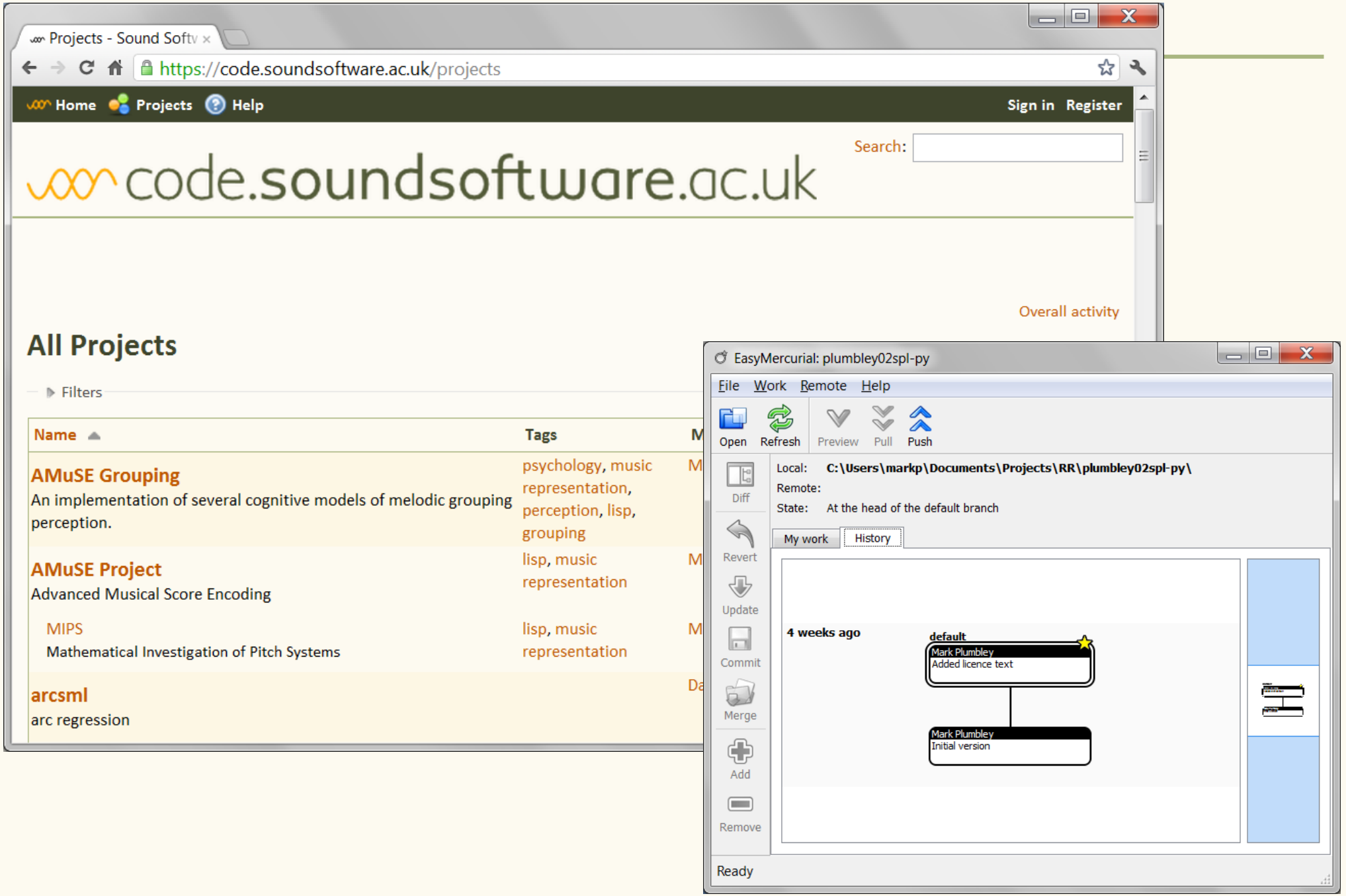
Our approach: code site: **http://code.soundsoftware.ac.uk**

- Focus on audio and music research
- Public and private projects
- Link publications with code

Also: User interfaces for version control

- Existing ones are surprisingly difficult

EasyMercurial: **http://easyhg.org** and tutorials and videos


Projects - Sound Softv
https://code.soundsoftware.ac.uk/projects
Home
Projects
Help
Sign in
Register
Search:
code.soundsoftware.ac.uk
Overall activity
All Projects
Filters
Name
Tags
AMuSE Grouping
An implementation of several cognitive models of melodic grouping perception.
psychology, music representation, perception, lisp, grouping
AMuSE Project
Advanced Musical Score Encoding
lisp, music representation
MIPS
Mathematical Investigation of Pitch Systems
lisp, music representation
arcsml
arc regression
EasyMercurial: plumbley02spl-py
File
Work
Remote
Help
Open
Refresh
Preview
Pull
Push
Diff
Revert
Update
Commit
Merge
Add
Remove
Local: C:\Users\markp\Documents\Projects\RR\plumbley02spl-py\
Remote:
State: At the head of the default branch
My work
History
4 weeks ago
default
Mark Plumbley
Added licence text
Mark Plumbley
Initial version
Ready

Barrier: Lack of incentive

Issue:

- Software not typically recognised as citeable or assessable research output
- Lacks publication conventions for authorship, makes academic rewards unclear

Our approach: Link publications to code on the code site

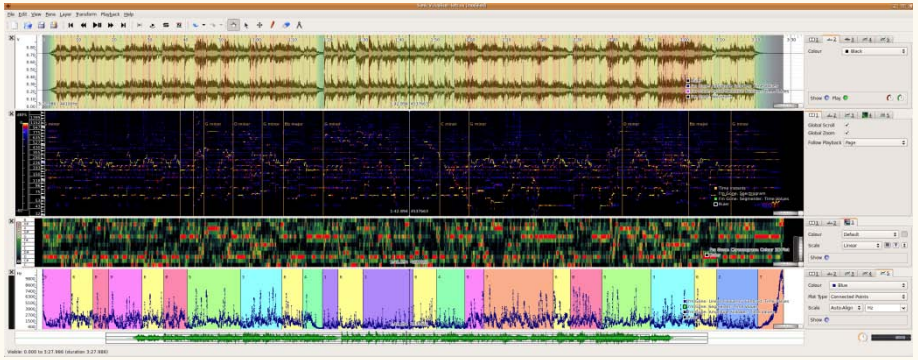
Purpose:

- Increase likelihood of code users discovering your papers
- Ensure users know how to cite your work
- Increase take-up / impact of your research

Barrier: Platform incompatibilities

Issue: Many different platforms and development tools in use

Some are not available to all possible users (e.g. MATLAB)



Our approach:

Use “plugin” approach if appropriate (e.g. SonicVisualiser)
produce standard code modules, reuse for multiple targets

Take maximum advantage of existing application ecosystems

Where do I start? (UK version)

1. **Register** an account on **code.SoundSoftware.ac.uk**
   - While you wait for account to be approved ...
2. Download and Install **EasyMercurial** from **easyhg.org**
   - Available for Windows, Mac OS/X and Linux
3. **Choose a software project** on your computer
   - For example, a set of files in one folder
4. Run **EasyMercurial**, **Open** the file folder, **Add** files, and **Commit**
   - This makes a "local repository" containing your project
5. On **code.soundsoftware.ac.uk**: Create a **new project**
   - Wait for the link. This makes a "remote repository" ready for your project
6. In **EasyMercurial**: **Push** your local project to the remote repository

Congratulations - Your software project is now under version control!

More things you can do

- Edit your code and **Push** updates to the remote repository
- Add other people you work with as **project members**
- **Merge** updates made by different people
- Create a **Release** of your code
- Make your project **Public** so others can use it

Where do I start? (Non-UK version)

You might have version control etc at own institution

- Does your research group/IT support provide it?
- Who should you ask? (We might be able to help)

Other alternative code repositories

- SourceForge, Google Code, GitHub, Bitbucket, ...
- Try our EasyMercurial from **easyhg.org**

See also: **http://soundsoftware.ac.uk/why-version-control**
and: "Choosing a repository for your software project" at
www.software.ac.uk/resources/guide
from our sister project, the Software Sustainability Institute

Suggestions for Research Groups

Aim at easy training targets

- Program structure, arranging code across files
- Tangible results for keen but inexperienced researchers

Provide / encourage version control and hosting

- Version control systems make an immense difference
- Use what you have available, or code.soundsoftware.ac.uk

Turn code into plugins

- Latch onto the existing ecosystems of popular applications

Encourage collaborative development

- Papers often co-authored, why not code?
- Create an environment of confidence about sharing

Conclusions

- Research too complex to be reproducible from paper alone
- Reproducible Research: Paper + Software + Data. Not Easy!
- Identify barriers and simple approaches to overcome them:
  - Lack of education / confidence with code -> Training
  - Lack of facilities and tools -> Provide repository and GUI
  - Lack of incentive for publication -> Link papers to code
  - Platform incompatibilities -> Take plug-in approach
- Version Control GUI – EasyMercurial (**easyhg.org**)
- For UK researchers: Repository **code.soundsoftware.ac.uk**
- If any UK interest - can we help your project? **Please ask us!**