How do I know that I have preserved software?

Brian Matthews,
Leader, Scientific Applications Group
STFC e-Science Centre

Arif Shaon, Juan Bicarregui, Catherine Jones, Esther Conway, David Giaretta, Brian McIlwrath
Science and Technology Facilities Council

- Provide large-scale scientific facilities for UK Science
  - particularly in physics and astronomy
- E-Science Centre – at RAL and DL
  - Provides advanced IT development and services to the STFC Science Programme
  - Strong interest in Digital Curation of our science data
  - Keep the results alive and available
  - R&D Programme: CASPAR, SCAPE, APARSEN
- STFC Interest
  - Keeping science data usable for long periods
  - Specialised scientific analysis software
  - Needs to be kept along with the data
  - s/w preservation projects
Work on software preservation

- **JISC projects (2007-09):**
  - Report on the Significant Properties of Software
  - Tools & Guidelines for the preservation of software as a research output

- **Software very large topic**
  - Diversity in:
    - `application of software`
    - `software architecture`
    - `scale of software`
    - `provenance`
    - `user interaction`

- **Project scoped to**
  - Scientific and mathematical software
  - Limited commercial consideration
  - Limit consideration of user interaction

Developed a framework for software preservation properties.
Software Preservation...

.... is worth doing
  – There are good reasons for us to try to preserve software

.... is complex
  – Software not easy to define.
  – Software has lots of different components and dependencies
  – Software operates in a complex environment

.... can be done in lots of different ways
  – Preservation, emulation, migration, cultivation, hibernation ...
  – Each strategy has different consequences and cost/benefits

How do we know:
  – What to preserve ?
  – Whether we have done it right ?
ICAT is a database with a well-defined API that provides an interface to Large Facility experimental data and will provide a mechanism to link all aspects of the research chain from proposal through to publication. ICAT is in use or development at several large facilities. For a full list see Facilities.

ICAT is developed as a collaboration between STFC eScience, STFC/ISIS Facility, Diamond Light Source and the ILL.

To browse the ICAT source code, visit the 'Source' tab.

If you have isolated a problem or want a new feature to be included in ICAT, please submit an issue.

Make sure to include all the relevant information when you submit the issue such as:

- ICAT version
- One line of issue summary and a detailed description
- And any workarounds if you have them.

The more information you provide, the quicker the issue can be verified and prioritized. A test case (source code) that demonstrates the problem is greatly preferred.

The ICAT code includes libraries under various compatible open-source license. A full list is available here.

- The ICAT Development Team
Software Preservation Steps

- What do we do when we preserve software?
  - Identify a number of related digital artefacts to preserve

- What do we do when we want to use it again?
  - Find the right software artefacts to use
  - Reconstruct them into a executable system.
  - Replay the execution of the system

How do I judge now that what I have preserved is “enough”?
What we need to support retrieval?

• Gross functionality:
  – Description of what the product does
  – Major input and outputs
  – Categorisation under a controlled vocabulary (e.g. GAMS)

• Ownership and legal control, licensing
• Provenance
• General software architecture principles
What we need to support reconstruction?

- Set of components and their dependencies
  - Including installation, configuration and build as necessary
- Programming language details
  - Compiler version if needed
- Specific operating system
- Specific hardware platform if needed
  - Including any dependencies on peripherals
- Specifics on required machine performance
  - RAM and disk space, processor speed, screen resolution
- Auxiliary libraries
- Auxiliary tools (with version)

May have versions and variants for different platforms
What we need to support replay?

- Detailed functional description
  - Input formats, output formats, API, error handling
- User Interaction model
- Programming languages
- Non-functional behaviour
  - Response speed, data size, security

**Environment**

- **Compilation, installation, configuration**
- **Software Execution**
- **Data Performance**

**Software Components**

**Data**
How do I judge now that what I have preserved is “enough”? 

• Based on the NAA performance model for digital preservation  
  – The test of the success of our preservation is the performance of the data for the user

• The Replay information are the Significant Properties  
  “those characteristics of digital objects that must be preserved over time in order to ensure the continued accessibility, usability, and meaning of the objects”

• Testing data performance to judge adequacy of the software performance.
Adequacy of Software Preservation

A software package can be said to perform adequately relative to a particular “significant property”, if in a particular performance it preserves those significant properties to an acceptable tolerance.

- Significant properties are evaluable features of the performance
- After the recall and reconstruction phase
- Supply **test cases** to evaluate the adequacy against properties
- Assesses the value of the replay
- **Can be generalised to any digital object**
Example Test Cases

- So tests on relevant significant properties need to be supplied
  - “the system should calculate the Fast Fourier Transform”
  - “the result must accurate to 8 decimal places”
  - “the pagination has to remain the same”
  - “the system should respond in 0.4-0.8 seconds”
  - “The user must be able to enter a 6 digit number representing a parameter”
  - “the user must be able to enter a 6 digit number representing a parameter in a text box on the upper left of the screen”
  - “the output should distinguish roads and rivers”
  - “rivers should be in blue (0000ff), roads in red (ff0000)”

- Need to supply test suites to prove that the preservation should be adequate on replay.
Significant Properties Editing and Querying for Software (SPEQS)

• Java-based Eclipse plug-in
• Enables capturing software preservation properties during its development
• Demonstrates the concept of preservation tools that could be integrated within existing software development systems
Preserving Legacy?

• Software repositories
  – Managed with preservation in mind
  – Provides capacity to capture relevant metadata
  – Encourage the right documentation

• PaNSoft
  – Software repository for Photon and Neutron data analysis software
Summary

• A framework for software preservation
  – Conceptual model of software
  – What components to preserve?
  – Preservation properties of software
  – Performance and adequacy
  – Fits in a OAIS compatible preservation methodology

• Validated in some practical scenarios
  – More validation and tools
  – In a methodology – risks and benefits
  – Integrated with a software repository.

• Ideally considered within a software engineering process
  – Good version control
  – Good documentation
  – Testing and test cases key to assuring adequacy of preservation.
Good software engineering leads to good software preservation.

http://sigsoft.dcc.rl.ac.uk/twiki/bin/view

http://www.e-science.stfc.ac.uk/projects/software-preservation/preserving-software.html

Thank You

Questions?

brian.matthews@stfc.ac.uk

http://www.e-science.stfc.ac.uk