Methodologies Involving Source Code Availability

The OAIS response to preserving an Access Software application execution service would likely depend, in part, on whether or not it had the source code for the application. If the OAIS had the source code and adequate documentation on the application, the expected approach would be to port the application to the new environment and attempt to test it adequately to ensure it was functioning correctly. As described in 4.2, it may not be obvious when the application runs but functions incorrectly. Ideally all possible output values would have been recorded initially so they could be used as the basis for ensuring correct functioning following the port. However, this level of testing is likely to result in an unacceptable cost/benefit ratio for the OAIS. Given that the application was compiled from original source code, it is probable that the algorithms are correct; the production of a test suite, or reuse of a test suite that was provided with the design documentation is probably adequate. As long as there is independent Representation Information for the Content Data Object, no migration need be involved.

If the Access Software was a proprietary package, which was widely used and available commercially, it is likely that there will be commercially provided bridge (i.e., conversion) software which Transforms the current Content Data Objects to other forms used by the new Access Software having a similar look and feel. This would be a Transformation type of Migration that is likely to be Non-Reversible. If no commercial alternative is seen, the OAIS may contract with the owner of the original Access Software to develop and provide source code for a simplified tool that can read but not modify instances of data written using the format. This would also be a Transformation type of Migration because of the change in software that is providing much of the Representation Information. This approach might not be viable because of cost or legal issues. In any of these cases, the OAIS will need to establish mechanisms to verify that agreed Transformational Information Properties are maintained. This requires that criteria have been established to clearly define what constitutes the Content Information as discussed in section 6. In addition the OAIS must investigate the issues of ensuring that the new Access Software is available to the Designated Community.

Potential Emulation Approaches

There may be a mandatory requirement from the Designated Community to maintain the look and feel of proprietary Access Software because of the large number of AIUs that are dependent on that Access Software. Proprietary Access Software will not have readily available Structure and Semantic Representation Information for the Content Data Object. In this case, if the OAIS is unable to obtain the source code, or has the source code but lacks the ability to create the required application for example because of unavailability of a compiler or operating environment, it may find it necessary to investigate use of an emulation approach.

The OAIS could consider emulating the application. If the application provides a well-known set of operations and a well-defined API for access, the API could be adequately documented and tested to attempt an emulation of the application.

One approach is emulation of the underlying hardware. An advantage of hardware emulation is the claim that once a hardware platform is emulated successfully all operating systems and applications that ran on the original platform can be run without modification on the new platform. However, the level of emulation is relevant (for example whether it goes down to the level of duplicating the timing of CPU instruction execution). Moreover, this does not take into account dependencies on input/output devices.

Emulation has been used successfully when a very popular operating system is to be run on a hardware system for which it was not designed, such as running a version of Windows™ designed for an x86 chip-set on a Unix system running on a different chip-set machine. However, even in this case, when strong market forces encourage this approach, not all applications will necessarily run correctly or perform adequately under the emulated environment. For example, it may not be possible to fully emulate all of the old hardware dependencies and timings, because of the constraints of the new hardware environment. Further, when the application presents information to a human interface, determining that some new device is still presenting the information correctly is problematical and suggests the need to have made a separate recording of the information presentation to use for validation. Once hardware??? emulation has been adopted, the resulting system is particularly vulnerable to previously unknown software errors that may seriously jeopardize continued information access. Given these constraints, the technical and economic hurdles to hardware emulation appear substantial.

There have been investigations of alternative emulation approaches, such as the development of a virtual machine architecture or emulation at the operating system level. These approaches solve some of the issues of hardware emulation, but introduce new concerns. In addition, emulation research efforts often involve a centralized architecture with control over all peripherals. The level of complexity of the interfaces and interactions with a ubiquitous distributed computing environment (i.e., WWW and JAVA or more general client-server architectures) with heterogeneous clients may introduce requirements that go beyond the scope of current emulation efforts. Emulating an operating environment with a small number of applications to be supported would make testing easier and appear to offer less risk of information loss. When emulations are developed to support Long Term preservation, their deployment is a logical extension of the Representation Information employed by the Access Software they support.