# 5th International Digital Curation Conference
### December 2009

## Assisted Emulation for Legacy Executables

Kam Woods and Geoffrey Brown

Indiana University

School of Informatics and Computing

December 2009

### Abstract

Emulation is frequently discussed as a failsafe preservation strategy for born-digital documents that depend on contemporaneous software for access (Rothenberg, 2000). Yet little has been written about the contextual knowledge required to successfully use such software. The approach we advocate is to preserve necessary contextual information through scripts designed to control the legacy environment, and created during the preservation workflow. We describe software designed to minimize dependence on this knowledge by offering automated configuration and execution of emulated environments. We demonstrate that even simple scripts can reduce impediments to casual use of the digital objects being preserved. We describe tools to automate the remote use of preserved objects on local emulation environments. This can help eliminate both a dependence on physical reference workstations at preservation institutions, and provide users accessing materials over the web with simplified, easy-to-use environments. Our implementation is applied to examples from an existing collection of over 4000 virtual CD-ROM images containing thousands of custom binary executables.

# Emulation and Access

Emulation is a key strategy for ensuring long-term access to born-digital materials, yet it has a vulnerability that is rarely discussed – its dependence upon original software and the corresponding assumption that future users will remember how to "drive." Even a seemingly simple task such as installation of software can be a formidable impediment to use.  We describe tools that will assist future users by automating tasks such as software installation that could otherwise compromise usability, as well as techniques to automate other common tasks, such as exporting data and generating reports using preserved software environments. The installation issues could be overcome by the alternative strategy of creating a custom emulation environment for each preserved object; however, space and software licensing concerns make this a relatively unattractive choice.  Furthermore, there remain usability issues that can only be addressed at runtime.

Our software provides users with access to legacy executables in automatically configured virtual machines using simple installation scripts. Scripts and relevant documentation are stored alongside the original digital objects and metadata. Our goal has been to address basic questions about preserving any contextual knowledge associated with legacy executables. We begin with simple automation of common installation and execution tasks. Additionally, we discuss "wrapping" older DOS-based and early Windows GUI applications to generate reports or export data. Automation allows users to quickly browse and obtain information from preserved programs much as they would traditional static documents, without the need to learn arcane installation procedures, risk contamination of the host environment, or manually reconfigure the virtual machine.

We show that with a small amount of coding - less than 600 lines of C# code to guide the user, configure an emulation environment, and run associated install scripts, and less than 100 lines of Java to link browsing of a web archive to the local application - we can create stable, repeatable environments for virtually any application type. Typical object specific install scripts are small (1-10 lines of code).

The work assumes that emulation is necessary to preserve some materials, and that the underlying technology is sufficiently well understood that preservation of emulation environments is likely to remain viable. We specifically address automating access to materials within an emulation environment, preserving and sharing emulation environments, and the technology required to automate emulation.

We believe digital archives with a mandate for open access must inevitably implement strategies of this type. Modern users are accustomed to convenient access to archival materials over the Internet, but do not necessarily have the expertise to reconstruct the environments necessary to use these materials (van der Hoeven, 2007). A survey of more than 4000 legacy CD-ROMs held in one collection at Indiana University identified more than 1900 unique binary executables distributed without source. These programs encode historically significant scientific, economic, legislative, environmental, and social data – data that is media-rich and frequently cannot be migrated or reprocessed into static documents without information loss.

Existing archival strategies and the software tools that support them have increasingly focused on considerations of future access - who will be looking for these materials, how they will find them, and under what conditions they will be used. Many of the assumptions made in these strategies are predicated on the idea that the majority of these materials will be stored as traditional documents - frequently word processing, spreadsheet, database, or presentation formats. Executables present more significant access and search problems. Modern indexing schemes are based primarily on document metadata and, to a lesser extent, on document content. As the complexity digital objects increases, facilitating end-user access becomes as critical a preservation problem as retention of the metadata used to describe them. Assisted emulation schemes such as the one presented here are a useful component in systems addressing this issue.

The remainder of the paper is organized as follows. We outline our approach, including basic layout of the client, server, and networked storage, and discuss the specific problems addressed by our software. We provide a detailed description of the operation of our software, including the adaptation of an existing online archive for both on-site and remote use. We describe preparation of the archive, including script generation for automated installation and data extraction. We examine existing emulation platforms and preservation-specific emulation approaches, and discuss some future directions for this and related work.

# Approach

Our approach focuses on improving user navigation of and access to programs located within a networked collection of CD-ROM ISO images presented as a single filesystem and accessible through a web interface (http://www.cs.indiana.edu/svp/). When a user clicks on a link to a virtual disk image for which an installation script is available, a Java applet executes a helper application on the user's workstation taking the form of a "wizard" that informs the user and provides the option of mounting the ISO image within a guest (Windows) virtual machine.
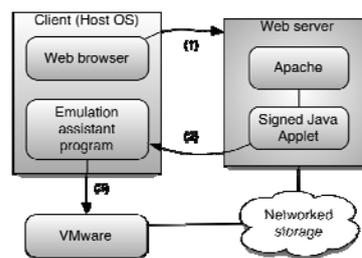


[Caption] Figure 1: Client request for networked resource.

Figure 1 illustrates the basic interaction between a user with an appropriately configured workstation and a remote site with legacy digital materials held on networked storage. A user navigating an online archive via a web browser locates a resource by entering search terms and selecting a desired ISO image from the result list. The user may browse the contents of the image directory structure in the browser, but for those images containing executables, two links are provided for direct access: one which executes the local helper application to configure a virtual machine using the appropriate network resource (.iso file mounted as a "virtual" drive), and one

which provides the option of downloading the entire image.

Installation scripts are stored alongside archival materials on logical volumes in a distributed, networked Andrew File System. Within each volume, .iso objects are stored in directories uniquely named according to a barcode assigned to that object. Installation scripts and additional metadata are stored alongside the CD-ROM images, as diagrammed in Figure 2. The software automatically polls a directory upon selection by a user to determine if an installation script is available.
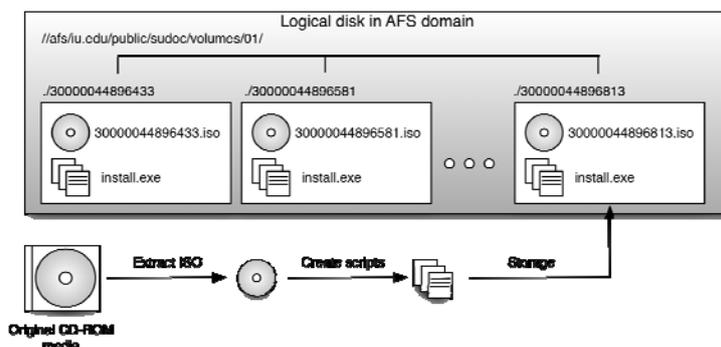


[Caption] Figure 2: Organization of the virtual archive.

There are several advantages to this approach. ISO images of CD-ROMs are large (typically hundreds of megabytes), placing considerable overhead on the network and subjecting users on typical network connections to a lengthy wait. When large resources like .iso files are stored in a distributed, networked filesystem such as an AFS, a mount of the resource as a virtual drive in a local VM enables network transfer of *only those parts* of the image that are required. Multiple users or virtual machines can access the images simultaneously. Finally, distributed filesystems that support a global namespace improve the ability of institutions and researchers to share, compare, and verify both data items and the environments used to support them.

Automated virtual machine configuration and networked resource storage eliminate the need for libraries to maintain physical "reference workstations", or hardware platforms with customized software environments to support access to legacy electronic materials. We focus in particular on executable software originally distributed on legacy media, including CD-ROMs and floppy disks. We envision future use profiles in which the experience of a user interacting with a particular collection using modern hardware is identical irrespective of their physical location or local software environment. With minor modifications, our software can accommodate any VM supported by the emulation software and any form of media capable of being mounted as a virtual device in the VM.

When interacting with legacy executable software, the user may find that lack of knowledge about an outdated operating system - or the time required to acquire that knowledge - hinders their ability to effectively browse the available materials. This issue has already arisen for older Windows and MS-DOS environments. What percentage of modern users browsing an archive are familiar with editing DOS-era batch files, or could diagnose silent installation failures due to a hardware incompatibility? Careful documentation and operational metadata can mitigate the problem somewhat, but eventually these instructions become little better than

incantations that the user must follow on faith, with little recourse should they fail.

Environmental integrity is another concern. A dedicated hardware workstation must either be preconfigured with every available piece of legacy software within an archive (a daunting task, not to mention problems associated with conflicting requirements and system security), or make available to the user installation and execution permissions that would lead to inevitable corruption or misuse of the environment. This can be overcome with the use of a virtual machine "snapshot" that is returned to a base configuration at the end of a browsing session, but the burden of manual installation remains with the user.

By providing a software-assisted method for automating interaction with the virtual machine, networked disk resources, and required software, we eliminate the problem of requiring extensive environmental knowledge from non-technical users. Our software can be readily deployed on common workstations, and the associated Java applet code can be used to link user activity in a browser to execution of a local virtual machine. The automation scripts are linked to individual ISO images and assume the presence of a "clean" virtual machine snapshot, effectively "freezing" the environment. This eliminates concerns of future environmental integrity or software changes. Additional scripts can be added to the archive as necessary without requiring alteration of the helper application.
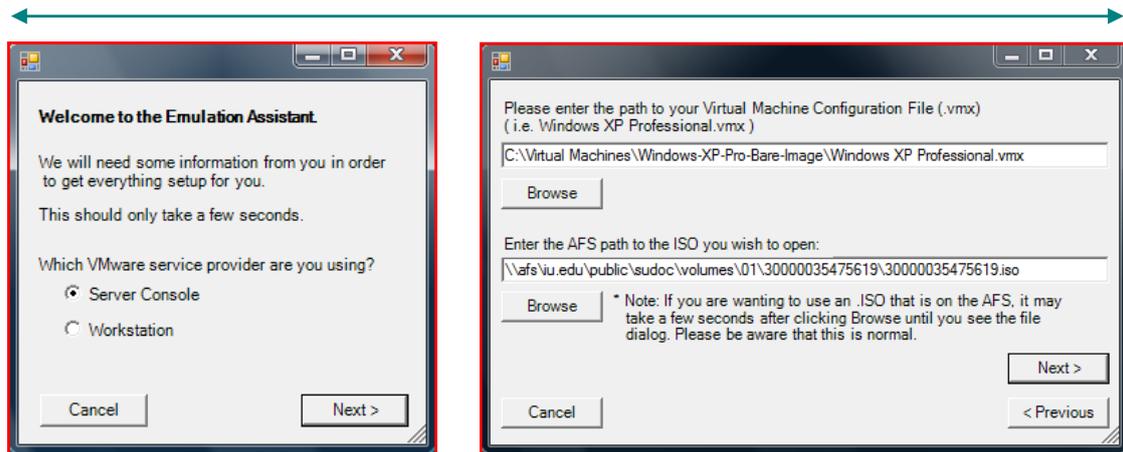
# Software

## *Overview*

Our software includes four basic components: a small "wizard" application to provide the user with assistance in configuration and execution of an emulation environment linked to a networked archival resource, a signed Java applet to execute the wizard on the workstation when the user navigates to a resource of interest on a given archival website, scripts to support installation and execution of legacy applications held on media images within the archive, and scripts to support data extraction via export functions within the applications.

The user selects a resource by clicking an appropriate link to a signed Java applet (running server-side), which in turn executes a helper application installed on the local workstation. This application first prompts the user to select the desired emulation package (Figure 3). For this implementation, we tested the software with VMware Server 1.0.2 and VMware Workstation 6.0.4, to ensure compatibility of the API calls with a variety of common products.

The user is presented with a final confirmation of their selections: the location of the virtual machine - the application automatically searches for paths corresponding to valid virtual machines and presents a default choice in an intermediate dialog - and the path to an ISO image corresponding to their original selection on the web site. The application (shown in this stage in Figure 4) may also be run in a "standalone" mode, with the user selecting a disk image manually.

[Caption] Figure 3 (left): Emulation platform selection. Figure 4 (right): Client request for networked resource.

Once these selections have been made, the software helper on the local workstation identifies the appropriate commands in the VMware configuration file, rewriting `ide1:0.filename` and `ide1:0.device type` to point to the appropriate ISO stored in the AFS. No further changes are required, although the application performs some basic tests to ensure that the user has not requested a missing resource. Finally, the application connects to the appropriate logical volume on the AFS to determine if any install scripts are available for the requested ISO. After a final user confirmation, the application starts the virtual machine from the existing snapshot, copies over any associated installation script, and executes it.

The ability to mount disk images as specific virtual devices, and beginning each installation in a "clean" virtual environment are critical components for successful access to many legacy executables. In our test archive, many of the associated installation procedures attempted to "update" the system with outdated library files. Manual installation attempts indicated that for a small but significant percentage of these installations, failure to install an older version of a software library broke the application. Additionally, many applications continued to expect to read resources (such as records from database files, spreadsheet tables, or maps) from the original CD-ROM mount point even after installation.

### *Installation Scripts*

Due to variability in installation routines and the fact that most legacy Windows executables are distributed as compiled binaries without source code, installation and configuration cannot be handled programmatically through the Windows API. Fortunately, a variety of high quality Windows GUI scripting languages exist for the express purpose of automating user-level tasks. We used the AutoIt tool, which provides a simple BASIC-style syntax for automation of the GUI. The following code (taken from a sample collection developed as part of our testbed) demonstrates automation of a simple Windows 95-era setup procedure:

```
Run("D:\Win95\SETUP.EXE")
WinWaitActive("Welcome", "&Next >")
Send("!n")
WinWaitActive("Choose Destination Location", "&Next >")
Send("!n")
```

```
WinWaitActive("Select Program Folder",
                "&Program Folders:")
Send("!n")
WinWaitActive("Start Copying Files","&Next >")
Send("!n")
WinWaitActive("Setup Complete","Finish")
ControlClick("Setup Complete","Finish",1)
```

The installations scripts are frequently even simpler than this, particularly for DOS-era programs for which only one or two commands are necessary, or for which the installation assumes a default path without informing the user. For those DOS-based applications that required additional changes to configuration and initialization files, help documents were included to describe the process of preparing the scripted installation.

Certain installation procedures depend on more subtle factors related to the operating system. Early Windows application may make (or require the user to enact) changes to system settings, such as those related to available fonts, drivers, or languages. These requirements may be opaque to future users. We encountered installation routines requiring the user to manually specify the maximum number of file handles available to the executable. In versions of Windows prior to 95, such changes were sometimes necessary to override system defaults set to conserve scarce resources. Scripted on-demand installation shields the end user from having to puzzle through historical technical requirements, without requiring the construction and maintenance of application-specific virtual machines.

We selected a set of approximately 150 CD-ROM images containing candidate executables for scripting from a list of more than 1900 custom executables identified in an original set of 4000 ISO disk images. 94 of the candidates had non-trivial installation procedures. A work-study student completed each installation in a "clean" Windows XP virtual machine, recording the necessary GUI actions, compatibility modifications, and (where necessary) alterations to the original installation procedure. Each action was recorded and encoded as an appropriate AutoIt command. 76 scripts were successfully completed. The remaining 18 exhibited compatibility issues with Windows XP. The successful scripts were subsequently compiled as executables and stored alongside the original archival materials in the networked Andrew File System disk array.
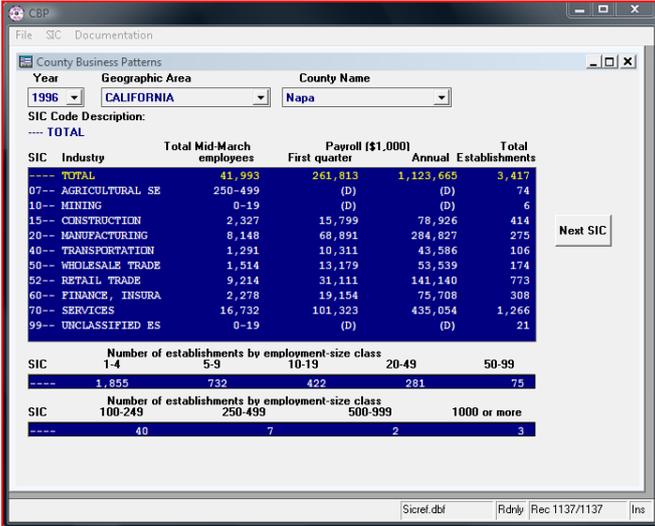
### Scripted Data Extraction

We observed that many of these legacy applications serve a single purpose – to reformulate data stored in a common file format (such as dBase or Microsoft Excel) and present the resulting "views" to the user. In one common case, raw data such as economic or census figures are stored in dBase files upon which various joins and filters are performed based on selections made by a user in a Windows 3.1-era GUI application. The number of available views is often limited, and the application provides a facility for exporting a columnar view as a text file.

While it is possible to migrate the dBase data directly to a more modern format, this may result in loss of context; without the data manipulation performed in the GUI application, the intended visual structure of the data (frequently non-trivial) can be

lost. Macro scripts such as those described in the previous section can provide a simple method for exporting structured information. In the example illustrated below, economic data published by the U.S. Census Bureau as a CD-ROM "County Business Patterns 1995-1996" is encoded in a series of legacy .dbf files that could readily be migrated to a more modern format using numerous open source and commercial software solutions. However, the data encoded in these files alone is not sufficient, as (for example) some field names are encoded numerically using the Standard Industrial Classification Code List, and only mapped to their natural language counterparts in the final GUI display. A sample of such a display is provided in Figure 5.



[Caption] Figure 5: Data view constructed from dBase files in a legacy Windows application.

For researchers interested in collating or searching data intended to be viewed using these executables, assisted data extraction provides high-quality, low risk views into the original format while requiring minimal interaction with the emulated environment.

### *Linking*

Because modern browsers operate in a "sandbox" designed to shield the user from malicious sites, we used a signed Java applet to link browsing activity with the local executable necessary to configure, boot, and perform required installations in a local virtual machine. Administrators can deploy this solution (or a modified one using the available source) simply by adding an applet-based link which points to the desired resource within existing pages. In our implementation, these links were generated automatically in the following form:

```
<applet code="RunExecutable.class" archive="RunExecutable.jar"
width="120" height="35">
<param name="exePath" value="C:\Legacy Emulation Assistant\Legacy
Emulation Assistant.exe">
<param name="afsPath"
value="\\afs\iu.edu\public\sudoc\volumes\04\30000038669341\3000003
8669341.iso">
</applet>
```

The Java applet reads two parameters: one corresponding to the path of the helper application on the user's workstation, and one linking the networked resource.  These can easily be customized; the applet exists only to provide a "trusted" link between the resource provider and the local workstation. Once executed, the local application itself can poll the system to determine what virtualization platforms are available.

### Emulation Platforms

In order to be effectively deployed in a production preservation setting, an emulation platform must implement features not only for technical preservation activities, but also for administration, configuration, and maintenance.  These facilities should be organized as management and automation APIs that can be used to normalize a virtual machine environment without concern for external factors such as where it has been deployed.

Emulation tools prepared by the digital preservation community remain largely primitive in terms of both technical and administrative features. Commercial products including VMware, Parallels, and CrossOver have set the standard for features we expect will be required in any large scale preservation solution, including robust APIs for 3rd-party development, support for a wide variety of virtualized I/O, network, and hardware devices, and automatic methods for deployment.

For this project we used VMware Workstation and VMware Server, along with the VIX automation API due to its relative stability and widespread use. However, our solution uses fewer than 100 lines of C# code to provide the needed automation support (start and stop of the VM, device mounting, and file operations), and could readily be ported to a fully open source solution such as QEmu. While configuration of products such as QEmu is somewhat less "user friendly", APIs such as libvirt[1] provide functionality similar to that available in commercial packages.

Existing commercial and open source virtualization products are stable and mature. They are routinely used in mission-critical environments to provide users with flexible access to customized environments independent of the underlying hardware. These products demonstrate the immediate potential for improved access to digital archives, and suggest that existing emulation platforms already meet the primary requirements of many organizations within the digital preservation community.

# Related Work

Much of the work on emulation in the preservation community has focused on questions that largely ignore the requirements of the end user. These include "Will a reliance on emulation adversely affect other preservation strategies?", "How can we ensure that the emulator itself is preserved?", and "What vulnerabilities are exposed when relying on imperfect emulation methods?". We believe that the issue of customization on demand – that is, whether we can reliably provide users with emulated environments to suit their needs on a per-resource basis, is of equal

---

[1] http://libvirt.org/architecture.html

importance. This is not just a *technical* issue, but one of building a sensible access requirements for existing and future archives.

In addressing the issues, the OAIS model makes specific reference to legacy Windows operating systems, and notes the following:

"[D]etermining 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 emulation has been adopted, the resulting system is particularly vulnerable to previously unknown software errors …"

The suggestion that a "reference recording" be retained to verify software operation is problematic; the vast majority of interactive software cannot be effectively verified this way except in the most rudimentary fashion. Additionally, legacy software for ubiquitous Microsoft operating systems is designed to run on a diverse range of hardware conforming to a compatible standard, to which a mature emulation platform such as QEmu or various VMware products (along with the requisite virtual machines, device drivers, and configuration options) comply.

Virtualization platforms developed within the digital preservation community are intended to be flexible, robust, reliably preserved, and readily executed on future architectures. Projects such as JPC and Dioscuri been developed with an emphasis on "write once, run anywhere" code – that is, compiled Java bytecode that can be run in a JVM irrespective of underlying architecture. These projects remain under development, but lack key administrative tools and APIs. A related approach, UVC-based emulation, has at least one implementation that plans support for dynamic executable content. (van Diessen et al., 2007)

Researchers have explored the use emulation in a variety of preservation frameworks (Lee, 2002). Some focus on the use of dedicated emulation environments to provide a high level of authenticity or closely replicate the "look and feel" of the original environment, particularly where historical and cultural factors associated with the software are significant (Mellor, 2003). Emulation is also frequently discussed in the context of preservation planning (Strodl et al., 2007). Until recently, however, there has been a dearth of published technical material on support for end-user interaction with emulated systems.

Larger research initiatives have examined not only the available virtualization tools, but also the formal requirements for their application in a wide range of environments. These include Preservation Layer Models developed for the IBM Digital Information Archival System (DIAS), and ongoing efforts to provide flexible access to a variety of emulation environments in client-server model such as GRATE (Global Remote Access to Emulation) as described by von Suchodoletz (von Suchodoletz, 2008) and more recently elsewhere (Rechert et al., 2009).

Additional evidence of the need for automated emulation environments linked to web-accessible archives comes directly from practicing scientists. This is exemplified by nanoHub[2], a distributed system enabling scientists to conduct simulations in software tools written for legacy platforms via a VNC client accessible via a website.

---

[2] http://www.nanohub.org

A software API is provided to generate an appropriate interface matching the original input and output requirements. The extensibility of this approach to more interactive applications may be limited.

# Discussion

We describe a software-assisted method to provide users with a simple and reliable method to install and execute legacy software in an emulated environment when browsing legacy archival materials. Our implementation has been extensively tested on the collection of legacy CD-ROM images available at http://www.cs.indiana.edu/svp/. The executables and source used to provide emulation assistance are freely available via SourceForge[3]. We have also made available the Java code used to generate the server-side applet for implementation at other sites. Automation and data extraction scripts for the materials in the SVP archive can be accessed via an AFS client pointed to the publicly available archival space.[4]

Our work is not intended to address issues of emulator preservation or selection of an emulation platform. Our focus is on demonstrating the feasibility of assisted emulation, particularly for archives with moderate budgets and technical expertise. The associated software is configured to interoperate with commercial virtualization products such as VMware that have mature APIs. The fundamental design – use of an open source distributed, networked filesystem, executable scripts constructed using a GUI-level macro language, and a simple Java applet to provide browser support – can be easily modified to support additional virtualization platforms and APIs.

This research describes simple software-assisted strategies to retain contextual information required to install, execute, and interact with legacy software. By automating both the process of configuring the virtual machine and of installing legacy software, we remove basic impediments to access that current and future users might face when browsing these types of materials. In future work, we plan to modify these tools to further assist in extraction of information from specialized legacy applications where traditional migration strategies limited.

# References

[report]Chang, W. (2006). *NIST Data Preservation and Migration Strategy: Virtualization*. Retrieved June 20, 2009, from http://www.itl.nist.gov/div8965/gipwog/Feb-2-06/NIST_DPMTest-Bed_at_GPO.pdf

[journal article]Lee, K.-H., Slattery, O., Lu, R., Tang, X., and McCrary, V. 2002. The State of the Art and Practice in Digital Preservation. *Journal of Research of the National Institute of Standards and Technology.* 107, 93--106.

[proceedings]Lorie, R. A. (2002). A Methodology and System for Preserving Digital Data. In *Proceedings of the Second ACM/IEEE-CS Joint Conference on Digital*

---

[3] http://www.sourceforge.net/aemtk
[4] //afs/iu.edu/public/sudoc/volumes

*Libraries, 2002, p312-319.*

[journal article]Mellor, Phil. CaMiLEON: Emulation and BBC Doomsday. *RLG DigiNews*. 2003 7 (2).

[report]National Archives (Netherlands). (2003). *Emulation: Context and Current Status*. Digital Preservation Testbed Whitepaper. Netherlands National Archives: Den Haag, June 2003.

[proceedings]Rechert, K., von Suchodoletz, D., Welte, R., van den Dobbelsteen, M., Roberts, B., van der Hoeven, J., and Schroder, J. (2009) *Novel Workflows for Abstract Handling of Complex Interaction Processes in Digital Preservation.* Proceedings of The International Conference on the Preservation of Digital Objects (iPRES 2009). San Francisco, October 2009.

[report]Rothenberg, J. (2000). *Using Emulation to Preserve Digital Documents*. (ISBN 906259145-0). Koninklijke Bibliotheek, NL.

[proceedings]Strodl, S., Becker, C., Neumayer, R., and Rauber, A. 2007. How to choose a digital preservation strategy: evaluating a preservation planning procedure. In *Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries* (Vancouver, BC, Canada, June 18 - 23, 2007). JCDL '07. ACM, New York, NY, 29-38.

[journal article]van Diessen, R. J., van der Hoeven, J. R., & van der Meer, K. (2005). Development of a Universal Virtual Computer (UVC) for long-term preservation. *Journal of Information Science*. 31(3) Thousand Oaks, CA: Sage Publications

[journal article]van der Hoeven, J. (2007). Emulation for Digital Preservation in Practice: The Results. *The International Journal of Digital Curation (IJDC). Beijing, China*: Bath: DCC

[report]von Suchodoletz, D. (2008). *Emulation Bridging the Past to the Future*. DPE, Planets and CASPAR Third Annual Conference: Costs, Benefits and Motivations for Digital Preservation. Nice, Italy. Retrieved August 1, 2008, from http://slideshare.net/DigitalPreservationEurope/emulation-bridging-the-past-to-the-future-dirk-von-suchodoletz-presentation

[proceedings]von Suchodoletz, D., and van der Hoeven, J. (2008). *Emulation: From digital artefact to remotely rendered environments*. In iPRES 2008: Proceedings of the Fifth Interna- tional Conference on Preservation of Digital Objects, 93– 98. The British Library, St. Pancras, London: The British Library.

[Internet resource]VMware, Inc. (2009). The VMware VIX API.  Retrieved June 20, 2009 from http://www.vmware.com/support/developer/vix-api/