Miles Levy

Digital Preservation

Professor Nicole Martin

Emulation: Preservation and Access Solutions

Computer emulation is a crucial tool for digital preservation in the broadest sense. The preservation of analog media artifacts related to moving image and audio recordings have long had standardized digitization and preservation workflows, best practices, and the expected advocacy related to the important task of preserving artistic and cultural heritage. However, preserving vintage software, video games, digital artworks, complex media installations, and computer hardware devices is inherently more complicated both technically, and often legally, than digitally preserving analog media. Creating a practical and legal template for the preservation of digital objects is still in its infancy on a mainstream institutional level in the very institutions which currently have comfortable, practical, and legal digitization workflows and best practices.

Most software, video games, and hardware were not created in an open source environment and thus were not created to be easily digitized or preserved. To add further complexity, computer software and hardware developers have planned obsolescence as a business model and with profit in mind. Preservation of older software, video games etc has largely been looked down upon on a logical level as companies want purchasers to constantly upgrade and move forward in the directions which these private companies spearhead. However, as we move further along in the digital age, the reality is that older software and their associated files must be preserved so that we have the ability to access files from associated software. As AV archivists, we must be concerned with the protection of all forms of media, including digital media, and not simply the more traditional moving image and audio components which are generally associated with the profession. This paper will attempt to broadly explain the different types of computer emulation and how institutions and individuals are approaching emulation as digital preservation. Before going into the history of computer emulation and its uses in the worlds of both the formal digital preservation/university libraries and artist communities, let us go over some of the basics of computer emulation for understandings sake. We will also look at alternative and similar/related methods to emulation such as migration, simulation, and the use of Virtual Machines.

Emulation is software that enables a computer system referred to as the *host* to behave like another computer system called the *guest*.[1]

> "The Essential idea behind emulation is to be able to access or run original data/software on a new/current platform by running software on the new/current platform that emulates the original platform."[2]

Emulation mimics the observable, or aesthetic, behavior of one software on a newer or different hardware or software device. Although the goal of emulation is to perfectly emulate the experience of the original software, on an ontological or forensic level, it is not necessarily

[1] https://en.wikipedia.org/wiki/Emulator

[2] Granger, Stewart (October 2010) *Emulation as a Digital Preservation Strategy*. Retrieved from http://www.dlib.org/dlib/october00/granger/10granger.html

exactly the same as the original software or hardware. Henceforth, an emulator allows the user access to a perhaps obsolescent file format by recreating the parameters necessary for the file to function correctly.

An alternative to emulation, migration, is broadly seen as an inferior solution to digital preservation as migration involves the "conversion, reformatting, or rewriting of the program code so that the preserved digital object is compatible with a current computing environment."[3] Rather than doing all of this work on a file by file basis, emulation recreates the necessary parameters to universally run any file or program that would work on any said operating system or on any said micro processer. For this reason, emulation is seen as a superior and more universal approach to combat file obsolescence. Migrating file formats rather than emulating applications also presents potential issues such as data loss of migrated files, or aesthetic differences in the look and feel of the file now being run on a different application than it was intended to use. With emulation as a solution, you create the exact experience of the original application and hence when a user inputs older files into an emulated application, the experience of the files/application will be as nearly identical as possible to the original aesthetic experience of running the file on its originally intended machine.

The two cardinal types of computer emulation are software for software emulation and software for hardware emulation. Although the goals of all types of emulation are to perfectly emulate the experience of the original software, on an ontological or forensic level, emulations are not necessarily exactly the same as the original software or hardware. Simulation, an alternate method to recreate computer programs would be the exact recreation of software on a

[3] Hedstrom, Margaret and Lampe, Clifford (January 2001) *Emulation Vs. Migration: Do Users Care*?. Retrieved from https://www.researchgate.net/publication/228737832_Emulation_vs_Migration_Do_Users_Care

forensic level. In addition to these categories, on a physical level another type of "emulation", hardware for hardware emulation which can be used towards the preservation of antique computers of complex media installations can be philosophically considered a type of emulation as well.

Before going into more detail about how emulation is achieved on a technical level, let's analyze an attempt at a complete emulation of one of the first interactive video art installations and how the three aforementioned forms of emulation were used in order to recreate the installation. Artists Grahame Weinbren and Roberta Friedman's interactive video art installation *The Erl King,* 1982, which featured what is now "obsolete hardware, artist-written software, and custom made components" [4] was recreated in emulated form in 2004 for New York's Solomon R. Guggenheim Museum exhibition *Seeing Double: Emulation in Theory and Practice.* An important exhibit for emulation advocacy, especially new media art installation:

> "This Exhibition tests the promise of an experimental treatment--emulation—for rescuing new media art from the ravages of time. *Seeing Double* features a series of original art installations paired with their emulated versions. This exhibition offers a unique opportunity for both art experts and the public to compare both versions directly and decide for themselves whether the recreations capture the spirit of the originals." [5]

Digital preservation expert Isaac Dimitrovsky undertook the task of emulating *Seeing Double* for the exhibition and through his emulation processes; we can illustrate through a categorical

[4] https://www.guggenheim.org/exhibition/seeing-double-emulation-in-theory-and-practice

[5] http://www.variablemedia.net/e/seeingdouble/

analysis the aforementioned three emulation strategies. Jeff Rothenberg economically summarized the installations functionality in a post-project report.

> "In a typical installation, a single "user" sits at a touch –sensitive color display screen, the visual and/or audio content changes in response. No menus or icons are displayed, and there is no keyboard or other input device aside from the touch screen"[6]

Further explaining the interactive nature of the early multimedia work:

> "Both video and audio may change at the same time, multiple video images may be superimposed, and text may appear on the screen, overlaying the video imagery. These interactive responses are controlled by a computer program that was designed by the artists and written by a small programming team." [7]

Jon Ippolito's video tutorial related to *The Erl King* helped to explain the various emulation strategies which were used for guidance in this analysis.[8] In order to recreate the look and feel of the original piece, but through the use of different physical components creative hardware for hardware emulation solutions were implemented. The CRT monitor was replaced by a flat screen monitor but was consciously made to appear identical to the original CRT monitor by encasing it in what was originally a wooden frame, but is now made of MDF rather than wood. When put side to side, visitors could not tell the difference between the monitors. The cabinet for the original installation was made of wood, yet they emulated this look by using modern MDF boards. To summarize, hardware for hardware emulation replaces physical

[6] Rothenberg, Jeff. (January 2006) *Renewing The Erl King.* Retrieved from http://archive.bampfa.berkeley.edu/about/ErlKingReport.pdf

[7] ibid

[8] Ippolito, Jon. *Three Types of Emulation.* Video tutorial retrieved at http://tutorials.nmdprojects.net/types_of_emulation/

elements of an AV installation so that it appears the same, while using different physical devices to achieve this goal.

The Erl-King mostly employs software for hardware emulation to achieve replication of the original installation. Generally speaking, software for hardware emulation is seen as a more stable emulation solution than software for software emulation as computers and their related hardware are designed to be quite stable and last for a longer period of time than any said software which is constantly being updated and changed. If one is able to emulate a piece of hardware through coding it into actual functionality, loading files into the hardware emulation will analogous to opening a file in an actual machine utilizing the actual on-board. Hardware emulation is a more direct pathway/solution than emulating software which is not supported by the modern machines chip architecture to work on the modern chip. Hence, once emulation code is written at the chip level, any application or file that would run on this chip would run on your newly coded emulated chip.

Highlighting software for hardware emulation solutions in *The Erl King*, in the original piece, three laser disc players were outputted into a multiport controller which was then inputted into a vintage Sony SMC-70 computer. For historical note related to AV preservation, although the “SMC-70 resembles a home computer, it was designed for professional video generation, for example in Cable television applications, and digital video effect generation.”[9] Below is an image of the SMC-70:

[9] https://en.wikipedia.org/wiki/Sony_SMC-70

Video output from the laser discs into a video switcher which alternated inputting video signals into the SMC-70 through control ports which were designed for audio cassettes used as external storage devices or alternately video material could be output directly into the CRT from the laser disc players. All audio signals were analog and were directly output into speakers and amplifier and then projected through speakers. The custom coded SMC-70 computer controlled the parameters related to the timing and control of different media options which were toggled by the user through the use of the touch screen.

When recreating *The Erl King* , a modern Linux computer replaced the SMC-70 and all aforementioned hardware components related to the AV material have been emulated in software for hardware emulations as well as the utilization of a Pascal interpreter, the original language used for coding, rather than implementing a software for hardware emulation of the SMC-70's Z80 microprocessor. This decision was largely made in relation to the fact that emulating the Z80 chip would be a waste of resources as there are relatively few computer-based artworks which run on this chip.[10]

[10] Rothenberg, Jeff. (January 2006) *Renewing The Erl King.* Retrieved from http://archive.bampfa.berkeley.edu/about/ErlKingReport.pdf p.16

The original source code for the installation was written in the MT+ Pascal computer language. This source code is human readable but too complex for a computer to interpret in real-time so as part of the recreation of *The Erl Kin,* an interpreter program rather than a software to software emulation of the original code was installed to translate the source code into a machine language object code which is expressed to the computer in a series of 1's and 0s.

Since the preservation goal of *The Erl King* was not an exact carbon copy replication of the device, but rather to create an experientially emulated aesthetic which would be indistinguishable on a phenomenological level to the user, emulating all aspects of the hardware or software verbatim was not necessarily the goal of the project. Although new hardware would be involved, whether emulated or physical, "the interactive experience for the user should be virtually identical to that of the original work."[11]

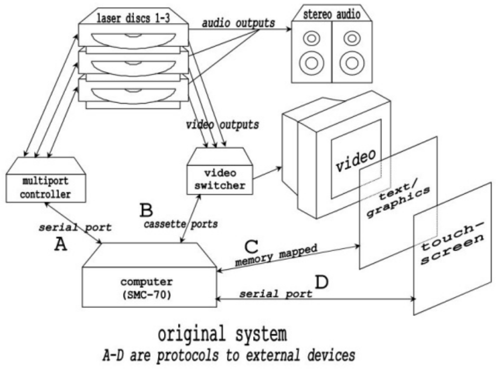
When planning for either alternative of the emulation of the original Z80 chip or the implementation of interpreted source codes it was approximated that either process would run extremely slow compared to original unmediated Pascal based computing processes. However, due to the modern Linux computers chip speed, being thousands of times as fast as the Z80 chip, this speed problem would be of inconsequence. The final decision was made to "emulate the external hardware devices, but interpret the original *Erl King* software at the Pascal code level instead of emulating the object code."[12]

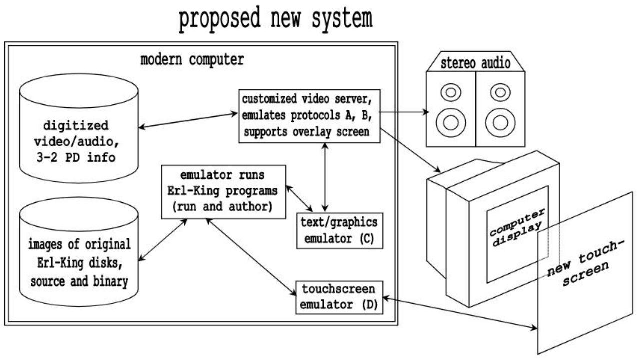
Software emulation was also employed in the process. The laser disc players software has been replaced by emulated software which runs video off of a hard drive. By emulating the

[11] ibid

[12] Dimitrovsky, Isaac. (April 1, 2004) *Final Report, Erl-King Project.* http://www.variablemedia.net/e/seeingdouble/report.html

original peripherals of the code which controlled video output, the recreation process reused the original code written by the original creators in a newly emulated software environment. Other emulations also involved software to hardware emulations. In the original piece there were three layers of screen. The CRT, a text/graphics overlay screen, and a touchscreen over the top. In the new configuration there is merely a flat screen and a touch screen over the top of that. Whereas before, a text graphics screen was output from the SMC-70, now a touch screen emulator was created which goes directly from the new CPU program to the touch screen. Further emulation allowed for the text/graphics screen controls to be emulated through a custom video server which supports the new overlay touch screen. The images from the original laser disc were digitized and coded to run through source and binary code through the new CPU and through the new customized video server. The below graphics illustrate the original system and the newly configured system which uses software for software and software for hardware emulations.




proposed new system
modern computer
digitized video/audio, 3-2 PD info
customized video server, emulates protocols A, B, supports overlay screen
stereo audio
emulator runs Erl-King programs (run and author)
text/graphics emulator (C)
images of original Erl-King disks, source and binary
touchscreen emulator (D)
computer display
new touch-screen

[13]

In his post-project report Isaac Dimitrovsky reflects that preservation through emulation is a tricky process and that there is no boiler plate emulation solution to recreating complex

[13] ibid

media installations. “In the case of preservation through emulation, many of the devilish details that cropped up here are likely to recur on similar projects.”[14] To Quote Dimitrovsky:

> “Each Preservation project will require its own set of techniques and compromises. Since some of these will not be anticipated in advance, I think the best process for this type of project would as a rule be a relatively free-flowing one involving developers, conservators, and if possible the original artists.”[15]

QEMU, Virtual Machines, and the EaaSI Software Preservation Network

We will now switch gears to software emulation and the use of virtual machines (VMs) as solutions to accessing older file formats and older software. This section will focus more on software to software emulation and virtualization than software to hardware emulation which was discussed in the above *The Erl King* analysis.

Briefly, the term “virtual machine” is a descriptive phenomenological term which describes the human user as virtually experiencing a computer as a unified thing and interpreting this virtualization as an actual isolated device such as a desktop or a laptop computer. Oftentimes VMs are created as a sort of partition of a hard drive in a larger system of servers. One can port into a larger server and experience a virtualized software environment and interact using actual files on their native computer and access them through the VM which is a shared piece of software in a larger network of software designed for sharing.

[14] Dimitrovsky, Isaac. (April 1, 2004) *Final Report, Erl-King Project.* https://www.guggenheim.org/wp-content/uploads/2015/11/guggenheim-conservation-treatment-report-erl-king-project-2004.pdf

[15] ibid

For example, a human would typicaly expect four computers to be four actual physical desktops. However, one could network four virtual machines to four computers and all four virtual machines would actually be derived from the same machine, but from partitioned portions of the one server. For references sake, all cloud machines utilize virtualization in order for users to access aspects of the cloud server and not the entirety of the cloud infastructure. To summarize, virtualization is more about management of hardware/software resources whereas emulation is more about overcoming obsolescence by running software which was never meant to be run on your modern hardware.

Switching gears to a specific program which utilizes virtualization of emulated software environments, QEMU is a networked application which allows for sharing virtualized environments which allow access to legacy operating systems and software. QEMU is used to share emulated software at various memory institutions emulation-as-a-service-strategies networks. QEMU is an open source machine virtualizer which emulates older operating systems on newer operating systems. The performance on QEMU is superior to other similar open source virtualizers by executing the guest code on the host CPU.[16] Henceforth, while it is still software emulation, it actually utilizes the guest's computer chip. On a modern 64 bit chip, this will lead to very fast and efficient computing through the VM. Since many, but not all, application are backwards compatible, if the emulated software can take advantage of using the same chip and not have to use another layer of emulation, such as replicating a chip, it is beneficial for speed's sake to use the actual CPU. Emulation software not having to do as much work is always a good thing.

[16] *What is QEMU?* (December 20, 2016) Retrieved from https://www.youtube.com/watch?v=OhUJeuviwhE&t=189s

In order to better understand how research libraries are utilizing shared software emulation, I spoke with EaaSI's Software Preservation Analyst, Ethan Gates. Ethan is currently working at Yale University's library as part of the EaaSI software preservation initiative. He provided a rundown of how and why EaaSI works and exists. Ethan explained that it takes a lot of work to set up emulated software. So, rather than repeating the difficult task of creating functioning emulations, the obvious choice would be to share previous emulation in a network of emulation libraries from around the world. Partner institutions, referred to as *nodes,* from around the world share libraries through servers and allow access through Yale's servers. Users tap into the network using QEMU as a mediator and QEMU creates a virtual machine from Yale's server for every authorized user of their shared emulated software. Disk images of emulated programs are sent between the nodes and Yale's server. When a user tape into the network through QEMU, a virtual machine is created temporarily with a limited amount of resources. Let's say the broader server has thousands of GB of ram cumulatively; the EaaSI network allows each user a small piece of this memory and allots it to be used on an ephemeral basis. When the user logs out of the system, the virtual machine immediately ceases to exist. This is different than the sort of virtual machine one would have access to from a permanent website server. That kind of virtual machine will permanently have a virtual machine set aside with whatever memory parameters have been agreed upon by the host and the guest.

The goal of EaaSI is to have a decentralized network. If one node were to shut down for whatever reason, only a portion of the projects library of emulators would go offline. Let's say for example, Stanford's library is sharing a vintage copy of Corel WordPerfect. If Stanford's network were to go down, no one would be able to use their emulated software, in this case WordPerfect. However, this is a superior solution for broad access as if everything were to be stored on one server, if that server were to fail the entire network would be unusable. Furthermore, for legal reasons, there are advantages to a decentralized shared network even if through fair use exemptions this sort of legacy software sharing seems to not be attracting negative attention from private software developers. The educational fair use exception has so far protected university emulation software sharing projects from suffering through negative legal interactions.

Ethan further spoke about his job duties on the EaaSI project and about his role as software preservation analyst. It seems pertinent to describe the role that AV archivists play within such software emulation sharing initiatives so let us delve into his role in the project to give perspective to other AV archivists who might be considering working in such an environment.

Ethan described his primary duty as ultimately helping users to properly experience their emulated software through EaaSI. It is his job to populate all emulated software into the network which is slated to open to users on March 1st, 2018. Yale received a large donation of imaged CD roms and floppy drives. Rather than getting official physical media off of a Windows 95 CD-ROM , their argument is these shared disk images are simply carriers not the actual physical media. There is a fair use argument to be made here that disk images were made to be used for education/preservation intents and will be accessed through a university log-in portal rather than simply "pirating" physical media and allowing the general public to use it. Google Books used a similar general educational preservation exception which has allowed them freely to distribute scanned versions of books online.

Ethan tests how software is running through the network and makes sure it is properly usable. He also oversees a team of student and staff whose job it is to properly catalog and archive all aspects of the project so it is findable in an intuitive manner. Ethan also creates documentation for the project such as user guides on properly installing the system, troubleshooting guides.

Lastly, Ethan is planning to run a backwards compatibility QC test to see if the emulated software from their system will actually run on the original legacy computers as if they were simply loading files through the application on the original vintage computer. His plan is to take disk images of the emulated software, place them on an external hard drive and plug this drive into a legacy computer. At this point he will test out the programs and see if they work, thus helping to conclude whether the emulations being used in the project go beyond simply aiding in access but could in fact be considered preservation copies.

In conclusion, computer emulation is an excellent tool which aids in the access and preservation processes of both software and hardware. Emulation, as well is migration, and simulation, can all aid in access and preservation initiatives and can work in harmony with each other or in isolation. It became apparent from the analysis of preserving and recreating *The Erl King,* that especially in regard to art installations utilizing vintage computing / AV hardware that customized rather than boilerplate solutions are necessary when approaching complex preservation projects. Like most aspects of computer science / engineering as well as preservation and conservation in general, the task at hand must be meditated upon before applying attempted solutions. Emulation is increasingly becoming an accepted method in the battle against obsolescence and it seems that AV archivists will only increasingly become involved with such projects as EaaSI and other software emulation sharing initiatives to come.