The International Journal of Digital Curation
Issue 2, Volume 2 | 2007

Emulation for Digital Preservation in Practice: The Results

Jeffrey van der Hoeven,
The National Library of the Netherlands

Bram Lohman,
Tessella Support Services plc., United Kingdom

Remco Verdegem,
The Nationaal Archief of the Netherlands

December 2007

Summary

In recent years a lot of research has been undertaken to ascertain the most suitable preservation approach. For a long time migration was seen as the only viable approach, whereas emulation was looked upon with scepticism due to its technical complexity and initial costs. In 2004, the National Library of the Netherlands (Koninklijke Bibliotheek, [KB]) and the Nationaal Archief of the Netherlands acknowledged the need for emulation, especially for rendering complex digital objects without affecting their authenticity and integrity. A project was started to investigate the feasibility of emulation by developing and testing an emulator designed for digital preservation purposes. In July 2007 this project ended and delivered a durable x86 component-based computer emulator: Dioscuri, the first modular emulator for digital preservation.

The *International Journal of Digital Curation* is an international journal committed to scholarly excellence and dedicated to the advancement of digital curation across a wide range of sectors. ISSN: 1746-8256 The IJDC is published by UKOLN at the University of Bath and is a publication of the Digital Curation Centre.

Introduction

Both the National Library of the Netherlands and Nationaal Archief of the Netherlands are responsible for the long-term preservation of an important part of the Dutch cultural, scientific and governmental heritage. The Nationaal Archief of the Netherlands has a legal obligation to preserve and give access to archival records of Dutch governmental organisations (OCW, 2002). The KB's mission is to ensure permanent access to information which also includes digitally stored information. With an increasing amount of digitized and born-digital documents and applications, new protocols and strategies for preservation and access are required. To secure storage of digital objects, the KB operates an electronic repository, called the e-Depot (KB, 2006), which by December 2007 contains more than ten million digital objects, varying from PDF documents to interactive multimedia applications. At this moment the Nationaal Archief of the Netherlands is developing a digital depot system which is expected to be fully operational by the end of 2008.

Long-term preservation of digital objects not only entails secure storage and management, but also includes the development and execution of strategies to secure sustained accessibility to these objects. Such strategies can roughly be divided into two groups: migration and emulation. Migration is focusing on the digital object itself, changing the object in such a way that software and hardware developments will not affect its original representation. By converting the format of an object, it is possible to render these objects on current systems. Emulation does not focus on the digital object, but on the hard- and software environment in which the object is rendered. It aims at (re)creating the environment in which the digital object was originally created.

The wide variety of digital formats and applications makes it impossible to choose a one-size-fits-all solution for preservation. Therefore, the choice for either strategy should be determined by the kind of digital object that needs to be preserved, the essential characteristics of the digital object, the requirements of a user that would like to access this object now and in future, and the policy of the institution responsible for preservation.

To make the choice for a preservation action more manageable, a digital object can be decomposed into five attributes: content, context, structure, appearance and behaviour (functionality) (Rothenberg & Bikson, 1999). All attributes together form the digital object as it is represented to the user. The importance of each attribute may vary depending on the requirements. If the original 'look and feel' of a WordPerfect 5.1 document is crucial, then the presence of the original WordPerfect application, version 5.1, is essential, in which case emulation is the best option.

Emulation in Practice

The choice for emulation as a preservation strategy is not undisputed, even though its benefits are recognized. Emulators like MS Virtual PC[1], QEMU[2] and Bochs[3], and virtualisation techniques used in VMware[4] have become very advanced and capable of running complete operating systems like Microsoft Windows, Apple's MacOS or

[1] Microsoft Virtual PC http://www.microsoft.com/windows/virtualpc/
[2] QEMU http://fabrice.bellard.free.fr/qemu/
[3] Bochs http://bochs.sourceforge.net/
[4] VMware http://www.vmware.com/

GNU/Linux, supporting a wide array of device peripherals. However, none of these solutions has been designed specifically for digital preservation. Emulation or virtualisation software that works today provides no guarantee that it will operate under different conditions in the future.

Various techniques have been proposed to overcome this problem (chaining or migrating emulators (Verdegem & van der Hoeven, 2006)), but these introduce a new risk of degrading functionality and performance each time the emulation process has to adapt to new circumstances.

From a preservation perspective this is an undesirable situation. The need for a digital preservation-proof emulator is evident, but requires high initial effort. There are claims that developing such an emulation solution is far too complex and expensive (Granger, 2000). Another drawback mentioned is that the lack of data exchange between the emulated and real environment is too much of a disadvantage to make it a worthwhile solution (Phelps & Watry, 2005).

As far as we know emulation has never been developed and tested within an operational digital archiving environment. The KB and Nationaal Archief of the Netherlands believe that emulation provides a good solution for long-term access to digital objects without affecting their authenticity and integrity and that this strategy has to be developed and tested first, before the potentials and limitations can be assessed.

In April 2005, the KB and Nationaal Archief of the Netherlands started a two-year project to develop a preservation strategy based on emulation. The objective of this project was to develop an emulator capable of recreating a modern x86 computer environment, while remaining durable and easy to configure. Furthermore, it should support a mechanism to transfer data between the emulated and real environment.

Design

Before the start of the development of the emulator, the KB conducted a preliminary study into emulation-based preservation. It explored the current state of the art and researched several possibile approaches to the creation of a preservation-proof emulation strategy (van der Hoeven, van Wijngaarden, Verdegem, & Slats, 2005). In cooperation with emulation advocate Jeff Rothenberg, this resulted in a new design for an emulation strategy: modular emulation (van der Hoeven, & van Wijngaarden, 2005). The principles of this strategy are based on earlier ideas about an Emulation Virtual Machine (EVM) of Jeff Rothenberg (2000) and the Universal Virtual Computer (UVC)-based preservation method of Raymond Lorie (2000). Two aspects distinguish this design from any other emulation approach: modularity and durability.

Modularity

The modular emulation strategy stays close to the basic architecture of today's hardware, known as the Von Neumann architecture[5]. Modular emulation can be defined as:

> Emulation of a hardware environment by emulating the

[5] Von Neumann computer architecture. Retrieved August 30, 2007 from http://en.wikipedia.org/wiki/Von_Neumann_architecture

> components of the hardware architecture as individual emulators and interconnecting them in order to create a full emulation process. In this, each distinct module is a small emulator that reproduces the functional behaviour of its related hardware component, forming part of the total emulation process. (van der Hoeven, & van Wijngaarden, 2005)

By applying such a component-based architecture, modules can be arranged in all kinds of configurations, similar to real hardware. Based on the requirements of operating system and applications, a customized emulator can be created that fits these requirements.

By emulating hardware, the original operating system, applications, drivers and configuration settings, which guarantee authenticity of the original software environment, are retained. Moreover, as hardware has to be manufactured, well-defined specifications are required. Although these are not always publicly available, they offer a better understanding of the hardware's functionality. Similarly, applied industry standards are well described and form a good starting point for learning about hardware interfaces.

Durability

Running software indefinitely is wishful thinking. Every computer application is dependent on its underlying platform consisting of system software and hardware. Changes in the platform hold implications for the software depending on it. Porting software to multiple computer platforms reduces the risk that none of these instances work over time. This line of thought forms the basis for creating a sustainable emulator and has been addressed in ideas of Jeff Rothenberg's EVM approach, Raymond Lorie's UVC and Olonys VM from Vincent Joguin (2006). These concepts propose an intermediate layer between host platform and emulator, called a virtual machine (VM). A VM supports the running of the same application on different computer platforms without the need to change that application. The only restriction is that the VM's interface to that application remains stable over time, while the interface between the VM and underlying host platform is adapted each time that platform changes. As a consequence, only the VM needs to be maintained over time, while the emulator that runs on the VM remains untouched. For this reason, the modular emulation strategy includes the use of a VM.

Modular Emulation Strategy

Figure 1 below depicts the system design for the modular emulation strategy. There are five main elements:

- Universal Virtual Machine (UVM)
- Modular emulator
- Component Library
- Controller
- Emulator specification document (ESD)

The UVM ensures that the modular emulator can operate on many different computer systems, now and in the future, ranging from a PC or Mac to a mobile device or embedded hardware. Configuration of the modules in the emulator is done via a controller. This component reads the desired configuration from an Emulator

Specification Document (ESD) which could be an XML-structured input file provided by a user or automated service. Using the information in the ESD, the controller selects the requested modules from a component library and starts the emulation process.

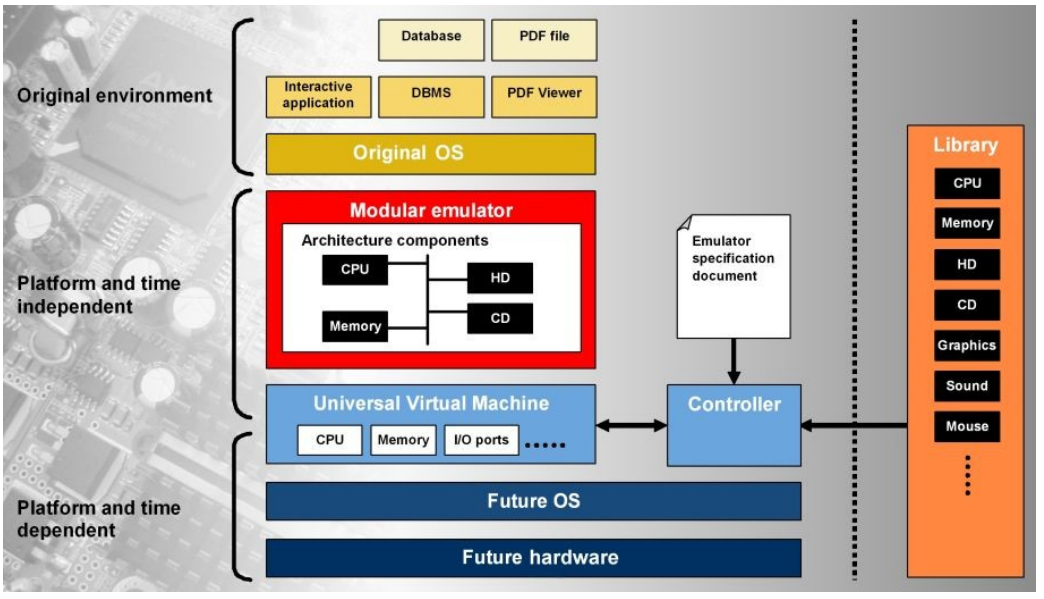

Figure 1. Design of modular emulation strategy.

Building the Emulator

In terms of basic functionality, it was decided to start by emulating an Intel 8086 CPU, the first of the x86 architecture - which is still the most popular today. Following the original development of the x86 architecture makes sense because all subsequent generations of the x86 architecture have been built on the groundwork of the 8086. Every x86 processor is backwards compatible, while the original 8086 as the core of today's x86 processors is still the same as that of the 8086 released in 1978 (Figure 2).

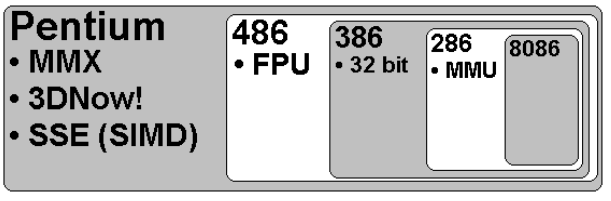

Figure 2. Incremental generations of CPUs.

This method of prototyping has another advantage: it allows a rapid development of the emulator, delivering several different models of a CPU while at the same time continuing development in the right direction.

Once the basic architecture was in place, other peripherals were added to interface with these components. The addition of proper input, such as the keyboard and mouse module, as well as proper output through a video card, along with a 'screen' to display the output, offers a huge amount of functionality to the emulator. Other, more indirect components such as an interrupt mechanism, interval timers, floppy and hard disk

support, have served to enhance the user experience of the emulator, and allowed for a more faithful rendition of the original system.

The development of these components naturally leads to a certain modularity, since each software module of a component is a small emulator by itself. However, with future iterations of the emulator in mind, careful selection of boundaries of each component must be taken into account, as the goal of the modularity is to provide the ability to interchange similar software components and still allow the emulator to run, within physical limits of course.

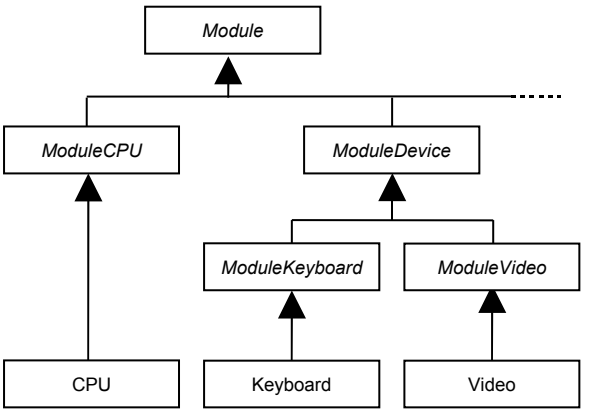

Figure 3. Modular structure in Java.

This requires a careful implementation of each component. Java was chosen as programming language because it is Object-oriented (OO) which supports the modular design and it provides a Java Virtual Machine (JVM), making the emulator portable to many Java-enabled platforms. For each hardware component, the logical switches and electronic pulses were translated into software objects with associated functions and interfaces. Using abstract classes, and a modular implementation where each component extends the parent class ('Module') and its inherited methods, Java guarantees that interchanging components will adhere to the standards set in the Module class (Figure 3). This way, any future module can be placed in an existing emulator and still be able to interface with other modules.

Reference Material

Building an emulator depends on the availability of accurate documentation, which is not always easy to come by. This problem is twofold: on the one hand, some of the components that are currently being emulated are more than 20 years old, and so a lot of the documentation has disappeared over the years. On the other, not all components were designed and based on industry standards, and this information was never published. A present-day example is the video card. Most major video cards are produced by companies for which commercial confidentiality is of the utmost importance. This means that no specifications are published, for fear of competitors learning trade secrets. This makes it especially difficult to emulate these components at a low level where documentation on the inner workings is vital but lacking.

Fortunately, other components that were developed according to official or de facto standards, and thus properly documented, can be easily emulated. The x86

architecture and ATA storage device interface are both widely published, and this information has made it fairly straightforward to build software components that fulfil the requirements.

Apart from openly available documentation, much advantage is derived from the open source community. Several open source emulators exist under the GNU General Public License (GPL) or similar, which provides freedom of information and sharing of code. The reuse of software has greatly benefited development, as there is no need to reinvent the wheel. This allows development to focus on other, less documented areas, and in turn share this knowledge with other developers. In this way, the community as a whole benefits.

Results and Testing

In July 2007 the first version of the modular emulator was released as open source software under the name Dioscuri[6]. It takes its name from the Greek myth of the twins Castor and Pollux[7]: one is mortal while the other becomes immortal. This is a symbolic representation of the idea behind emulation and long-term preservation: giving mortal digital objects their immortal equivalents. At the time of writing the latest release is version 0.2.0 which can emulate the functionality of the following hardware components:

- Intel 8086 based (16-bit) Central Processing Unit (CPU)
- Random Access Memory (RAM)
- VGA graphics adapter
- Text and graphics display
- Floppy disk support
- Hard disk support
- Interrupt handling
- Timing mechanism (clock)
- BIOS and CMOS settings
- Motherboard supporting I/O address space
- Direct Memory Access (DMA) for fast data transfer
- AT/XT/PS2 compatible keyboard supporting multiple keyboard layouts

For configuration of the modules a built-in controller and graphical user interface (GUI) are offered. The configuration is stored as an XML-based Emulator Specification Document (ESD). At the time the emulation process is started, the ESD is loaded and the modules are selected, initialized and, if necessary, connected. Following a hard reset, the emulator starts execution by loading the BIOS in memory and performing a Power-On Self Test (POST). Based on the user-defined boot sequence, Dioscuri is able to execute data from a floppy image or hard disk image (no physical media is accessed).

Execution and portability tests

Version 0.2.0 of Dioscuri successfully runs a BIOS and various versions of MS-DOS (4.0, 5.0 and 6.2), of which an example is shown in Figure 4 below. Using MS-DOS as their operating system, many 16-bit applications are also able to function correctly. Applications like Norton Commander 3.0, WordPerfect 5.1, DrawPerfect 1.1

[6] Dioscuri. Retrieved August 30, 2007 from http://dioscuri.sourceforge.net
[7] Greek myth Dioscuri. Retrieved August 30, 2007 from http://en.wikipedia.org/wiki/Dioscuri

and old games like Chess, Ironman and PC versions of Tetris and Prince of Persia all work well. A small number of other applications still report some missing CPU instructions, but this can easily be overcome by implementing these instructions in newer versions of the CPU module.

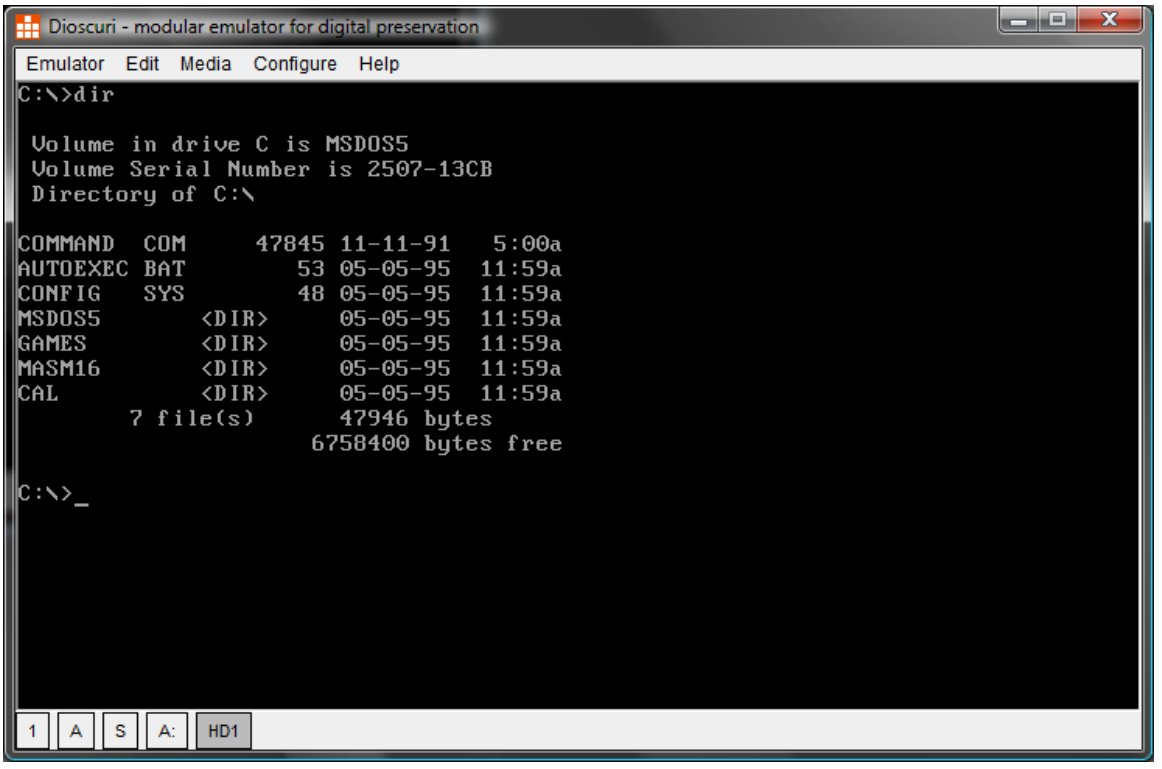

Figure 4. Screenshot of MS-DOS 5.0 running on Dioscuri.

Interesting results were obtained by comparing applications running on Dioscuri with the same applications executed directly on the host machine. In one of the tests, Norton Commander 3.0 was executed both on Dioscuri running MS-DOS 5.0 and directly on Windows XP. Although the application behaved similar, the appearance was slightly different. Some ASCII characters seemed to have been substituted by different characters on Windows XP. Although the text was still readable, this is obvious proof of the kind of information that would be lost without emulation.

Aside from running MS-DOS, Dioscuri is also capable of running FreeDOS 0.9 Beta (an open source version of MS-DOS[8]) and the 16-bit Linux operating system ELKS (Embeddable Linux Kernel Subset[9]). Support for these operating systems shows the versatility of Dioscuri.

The portability of Dioscuri becomes visible when executing it on various computer systems without any change to the software. Using the pre-installed Sun Java Runtime Environment (JRE) version 1.5.x, Dioscuri successfully runs on an Intel Pentium 4 running Windows XP, Vista and Linux Kubuntu 7.04, a Sun Sunblade 150 running Solaris 8, and an Apple Macbook Pro running Mac OS X 10.4.10. In all cases Dioscuri was able to emulate MS-DOS version 5.0 with its native look-and-feel.

[8] FreeDOS 0.9 Beta http://www.freedos.org/
[9] ELKS http://elks.sourceforge.net/

Data Extraction

Early results show that text can be copied from within the emulated environment to the clipboard of the host computer. Text can be pasted in any normal application running directly on the host computer. This method of data extraction shows that the emulated environment is able to communicate with its host environment via a common interface. Although extraction is limited and data insertion is not yet supported, it offers new possibilities for using emulation as a preservation action. Information contained in an old format could be viewed under emulation and extracted into a modern environment.

Conclusions and Future Perspective

Although developing an emulator is not an easy task, the joint project of KB and Nationaal Archief of the Netherlands has shown that it is feasible, even with limited resources. The total effort is approximately two man-years. Much work still has to be done, but the current version of Dioscuri already shows its value by executing old applications more accurately than is done by a modern computer platform. Dioscuri is more durable than other emulators as it is portable to a great variety of computer platforms without extra effort. Due to its modular design, Dioscuri can be configured into any target platform based on the available modules. Even so, data extraction makes it easy to transfer information between the emulated environment and the outside world.

Having reached this milestone, next steps are already in progress. Since July 2007 Dioscuri officially became part of the European project Planets[10]. Planets is a four-year project with the primary goal of building practical services and tools to help ensure long-term access to digital cultural and scientific assets. Within this context development of Dioscuri is being continued in order to extend its functionality. The processor will be extended to support 32-bit computing, allowing execution of a greater range of software like modern versions of Microsoft Windows, Linux, and applications. Support will be added for new modules like mouse, sound and improved graphics. Also, a module library will be designed and the configuration and invocation of the emulator will be further automated.

By making Dioscuri open source and supporting a dedicated emulation development platform, the project has ensured that any interested individuals or organisations can use Dioscuri for their personal needs or integrate it into their business process. Furthermore, this will, it is hoped, stimulate joint development that will take Dioscuri even further.

References

Granger, S. (2000, October). *Emulation as a digital preservation strategy. D-Lib Magazine, Vol. 6* (10). Retrieved August 30, 2007, from http://www.dlib.org/dlib/october00/granger/10granger.html

Joguin, V. (2006). Emulating emulators for long-term digital objects preservation: the need for a universal machine. (Emulation Expert Meeting 2006, The Hague, The Netherlands). Retrieved August 30, 2007, from

[10] Planets http://www.planets-project.eu/

http://www.kb.nl/hrd/dd/dd_projecten/slides/eem_aconit_vjoguin.pdf

KB (Koninklijke Bibliotheek). (2006). e-Depot, Koninklijke Bibliotheek, The Hague, The Netherlands. Retrieved August 30, 2007, from http://www.kb.nl/dnp/e-depot/e-depot-en.html

Lorie, R.A. (2000). *Long-term archiving of digital information.* IBM Research report, IBM Almaden Research Center, San Jose, Almaden.

OCW (Onderwijs, Cultuur en Wetenschappen). (2002). Regeling Geordende en toegankelijke staat archiefbescheiden (Regulation on the Arrangement and Accessibility of Records, February 2002). Nationaal Archief: The Hague, The Netherlands. Retrieved August 30, 2007, from http://www.nationaalarchief.nl/images/3_2563.pdf

Phelps, T.A., & Watry, P.B. (2005). A no-compromises architecture for digital document preservation. In *Proceedings from ECDL 2005, Vienna, Austria, 2005*, p. 266.

Rothenberg, J. (2000). An experiment in using emulation to preserve digital publications. Koninklijke Bibliotheek: The Hague, The Netherlands, pg. 8.

Rothenberg, J., & Bikson, T. (1999). Digital preservation – carrying authentic, understandable and usable digital records through time. Digital Preservation Testbed: The Hague, The Netherlands, p. 46.

van der Hoeven, J.R., & van Wijngaarden, H.N. (2005). Modular emulation as a long-term preservation strategy for digital objects. *Proceedings of IWAW, Vienna, Austria, 2005*. Retrieved August 30, 2007, from http://www.iwaw.net/05/papers/iwaw05-hoeven.pdf

van der Hoeven, J.R., van Wijngaarden, H., Verdegem, R., & Slats, J. (2005). Emulation – a viable preservation strategy. Koninklijke Bibliotheek / Nationaal Archief: The Hague, The Netherlands. Retrieved August 30, 2007, from http://www.kb.nl/hrd/dd/dd_projecten/Emulation_research_KB_NA_2005.pdf

Verdegem, R., & van der Hoeven, J.R. (2006). Emulation: To be or not to be. In *Proceedings IS&T Archiving Conference, Ottawa, Canada, 2006*, pp. 56-60.