SEVEN STEPS FOR RELIABLE EMULATION STRATEGIES SOLVED PROBLEMS AND OPEN ISSUES

Dirk von Suchodoletz
Klaus Rechert
University of Freiburg
Institute of Computer Science

Jasper Schröder
IBM Netherlands
Amsterdam

Jeffrey van der Hoeven
Koninklijke Bibliotheek
The Hague

ABSTRACT

After four years of research within the PLANETS project and two years of KEEP the jigsaw puzzle of emulation becomes a more complete picture. Emulation strategies are now seen as a viable complement to migration. A conceptual and theoretical groundwork has already been laid out, e.g. proper definition and selection of suitable emulators. However, integration into preservation frameworks and additional software archiving remain open research questions. This paper discusses several aspects of reliable integration and proposes development steps for a more complete emulation-based strategies in long-term preservation.

Introduction

For more than fifteen years there has been a vital debate on using emulation as a strategy to ensure long-term access to digital records. Although emulation has always been an essential addition for many types of digital objects, emulation strategies still have little relevance in practice despite many shortcomings, such as improper handling of dynamic artifacts and authenticity problems in various migration strategies. In contrast to migration, emulation does not require changes to the object or its structure. Hence, the original state of the digital artifact and its authenticity is preserved. However, emulation strategies are considered too expensive and too complex to be a viable solution to address digital preservation challenges [1].

Research on emulation as a long-term archiving strategy matured since the first reports on archiving of digital information in 1996 [7], fundamental experiments with emulation executed by Rothenberg [10] and the theoretical and practical work within the longterm preservation studies of IBM and the Netherlands National Library [13]. The Keeping Emulation Environments Portable project [1] aims to develop a strategy that ensures permanent access to multimedia content, such as computer applications and console games. The main research focus is on media transfer, emulation and portability of software. The platform will allow an organization to capture data from old physical carriers and render it accessible to users by using emulation. [2] To avoid the platform itself from becoming obsolete, a virtual layer guarantees portability to any computer environment [4].

Up to now there has been a strong focus on different emulation concepts as well as strategies to preserve the emulators themselves [14]. [3] Especially if looking at the more general emulation approaches, the question of additional software components needs to be taken into consideration (Fig. 1). Additionally, some relevant factors like the integration into emulation frameworks and the cost-effective application of emulation were ignored.

This paper gives an overview on the current status of research and describes requirements and challenges for successful emulation strategies. Therefore, we present a number of solutions like automation of emulation sessions, migration-through-emulation workflows and suggestions of preservation framework integration.

Step 1: Software Instead of Hardware Museums

An obsolete hardware collection is not a viable solution to preserving old computer architectures. The only reason for keeping hardware is to enable access to deprecated media for digital archeology or to present old platforms in a specific setting like a technical or computer games museum. The number of items to preserve risks becoming too large as it is a necessity to preserve environments for various types of digital artifacts. In addition, the space needed for a larger number of devices together with the energy required to educate and employ a very specialized maintenance crew for every different system would be a

[1] KEEP, *http://www.keep-project.eu*

[2] Requirements and design documents for services and architecture of emulation framework *http://www.keep-project.eu/ezpub2/index.php?/eng/content/download/7918/39623/file/-KEEP_WP2_D2.2_complete.pdf* Specification document for all layers of general-purpose virtual processors, *http://www.keep-project.eu/ezpub2/index.php?/eng/content/download/7917/39619/file/-KEEP_WP4_D4.1.pdf.*

[3] Emulation Expert Meeting 2006 in The Hague, *http://www.kb.nl/hrd/dd/dd_projecten/projecten_emulatie-eem-en.html.*

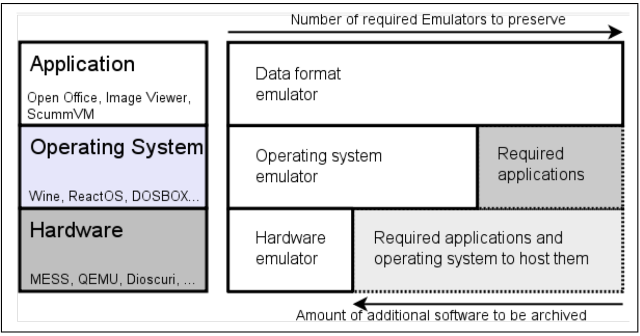
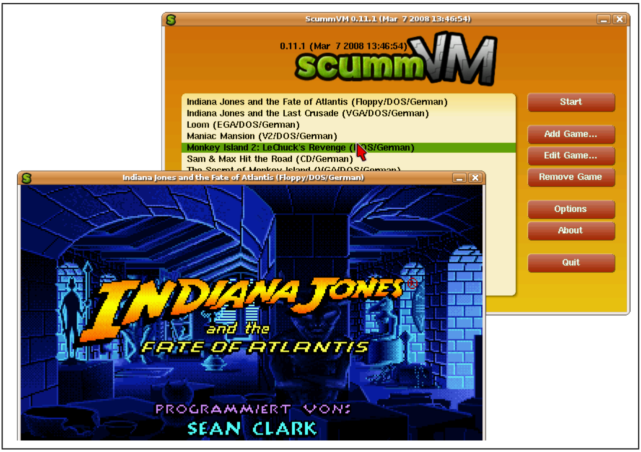

Number of required Emulators to preserve
Application
Open Office, Image Viewer, ScummVM
Operating System
Wine, ReactOS, DOSBOX...
Hardware
MESS, QEMU, Dioscuri, ...
Data format emulator
Operating system emulator
Required applications
Hardware emulator
Required applications and operating system to host them
Amount of additional software to be archived

Figure 1. Number of emulators and amount of additional software required depending on the layer chosen


ScummVM 0.11.1 (Mar 7 2008 13:46:54)
0.11.1 (Mar 7 2008 13:46:54)
scummVM
Start
Add Game...
Edit Game...
Remove Game
Options
About
Quit
Indiana Jones and the Fate of Atlantis (Floppy/DOS/German)
INDIANA JONES and the FATE OF ATLANTIS
PROGRAMMIERT VON:
SEAN CLARK

Indiana Jones and the Fate of Atlantis (Floppy/DOS/German)
Indiana Jones and the Last Crusade (VGA/DOS/German)
Loom (EGA/DOS/German)
Maniac Mansion (V2/DOS/German)
Monkey Island 2: LeChuck's Revenge (DOS/German)
Sam & Max Hit the Road (CD/German)
The Secret of Monkey Island (VGA/DOS/German)

Figure 2. ScummVM is a popular application level emulator for the Lucas Arts and similar type of games

large feat [3].[4] Unfortunately, electronic circuits will not run forever. They will fail at some point independent of the usage pattern. Furthermore, the probability of finding a spare part becomes slimmer each year a platform is out of production. Finally, the concept is fully dependent on the location, meaning there is no easy way to share resources between different memory institutions and users might have to travel large distances for access to a certain system.

Emulation uses a different approach compared to other well-established migration strategies in digital preservation. Emulation strategies usually do not operate on the object itself, but are intended to preserve the object's original digital environment. Emulation helps in becoming independent of future technological developments and avoids the modification of certain digital artifacts in a digital long-term archive.

The concept of emulation is not new to computer science. Emulators have existed for quite some time. Therefore, the list of the developed emulators is astoundingly long and covers a fairly wide range of areas. Prominent examples in the Open Source community are projects like ScummVM (Fig. 2), QEMU, Mess or Mame, just to mention a few. Not every emulator, however, is suitable for the needs of a long-term digital archive. Requirements of the respective archiving organization need to be differentiated, e.g. a national archive requires different computer platforms than a computer games museum. Emulators preserve or alternatively replicate old digital environments in software. They bridge outdated technologies with modern computer environments. Generally, for current computer platforms, three levels for the implementation of emulators can be identified: Topmost the application layer, followed by the operating system layer and on lowest level the hardware layer (Fig. 1). The latter uses the broadest approach, meaning no application and operating system needs to be rewritten to be able to access thousands of different digital object types. The function set of a hardware platform is straight-forward and often much smaller compared to operating systems or applications. Another advantage results from the smaller number of hardware platforms in comparison to operating systems.

[4] Computer Museum Universiteit van Amsterdam, *http://www.science.uva.nl/museum*

Step 2: Preserving the Emulator

Emulators face the same problems as do every software package and general digital objects. For this reason the considerations of perpetuation of the emulator for future use is a central component of a reliable preservation strategy [15]. Hence, emulators need to be adapted to the current hardware and operating system combinations regularly. The possibility of software migration is achieved through a suitable choice of emulators. If the emulator is available as an Open Source package, it can be ensured that a timely adaption to the respective new computer platform appears. Common and portable programming languages such as C should allow a translation with the respective current compiler. The main advantage of this approach is the use of only one emulation layer.

If there is no possibility to port the emulator on to a new host platform, the recently outdated host platform for which the emulator was created can be emulated [14]. This is referred to as nested emulation. This is a considerable advantage to avoid the complexity of a migration approach.

In the field of hardware emulation and virtualization (e.g. x86 architecture), successful commercial as well as Open Source solutions co-exist. During PLANETS we observed that commercial solutions, like VMware, Parallels of VirtualPC are not suitable for long-term horizons, since the vendors merely follow short-term interests, e.g. support current operation systems and software. With QEMU and Dioscuri two valid open alternatives exist.

QEMU is a multi platform Open Source emulator implementing x86, ARM, Sparc, PPC and further architectures. It supports a wide range of peripheral hardware from every era of the different platforms. We closely observed the development and advancements of the project throughout the duration of PLANETS and recorded significant advancements. Nevertheless, a number of problems like volatile support of major operating systems occurred and needs to be taken into consideration.

Dioscuri None of the mentioned emulators have been developed primarily for the purpose of long-term archiving of digital objects. This has changed with current research. Dioscuri [11] is a modular emulator which supports both recreation of an x86 computer environment and a durable architecture. With such a design Dioscuri is capable of running on many computer systems without any changes to the software itself. That way, there are chances that the emulator will sustain. At current state Dioscuri can render all kinds of applications from the MS-DOS era. The emulator is developed in Java which runs on top of the Java Virtual Machine and thus is portable to any computer platform that has a JVM running. The internal structure of Dioscuri is very similar to that of common hardware. Each functional entity (e.g. CPU, memory, storage, graphics) is implemented as a software module. Configuring these modules creates a virtual computer.

UVC In the course of research of the last few years we investigated alternate approaches like Universal Virtual Computer [6, 12]. UVC is different as it specifies a computer which is generally available.[5] The intention is to keep the specification of UVC stable over a long period of time. The instruction set of the UVC is limited and during the PLANETS project two new implementation strategies have been developed, one in C++ and one in C. The development effort for each version consisted of roughly four months of work. The expectation is that in the future a new implementation of the UVC can be made in a reasonable amount of time. Having a stable virtual computer layer, preserving the operating system and the applications on top of that ”hardware layer” (Fig. 1) will not be a big issue. These layers will keep doing their jobs. The complexity of the UVC-preserved applications has increased. In the beginning just image conversion applications were available on the UVC. During PLANETS, the logic of the SharpTools spreadsheet was ported as an example for more complex logic. As emulation is no longer a standalone activity and the UVC was made available as a Web service.

Step 3: View Paths

Digital objects cannot be used by themselves, but require a suitable context to the already mentioned working environment in order to be accessed. This context, called working or utilization environment, must combine suitable hardware and software components so that its creation environment or a suitable equivalent is generated, depending on the type of the primary object. No matter which emulator is chosen, contextual information of the original environment of the digital artefact was created in is always required. For example, questions such as ”for which operating systems is Ami Pro 3.0 compatible with?” are less obvious today then twenty years ago. To overcome this gap of missing knowledge, a formalization process is needed to compute the actual needs for an authentic rendering environment. In 2002 the concept of a view path [6] was proposed which we refined during research on emulation in PLANETS [15, 16]. A view path reproduces old computer environments or corresponding equivalents as ways from the object of interest to the working environment of the archive user. In other words, a view path is a virtual line of action starting from the file format of a digital object and linking this information to a description of required software and hardware (Fig. 3). Depending on the type of object, a specific rendering application may be required. This application requires a certain operating system to be executed, whereas in turn, it relies on particular hardware.

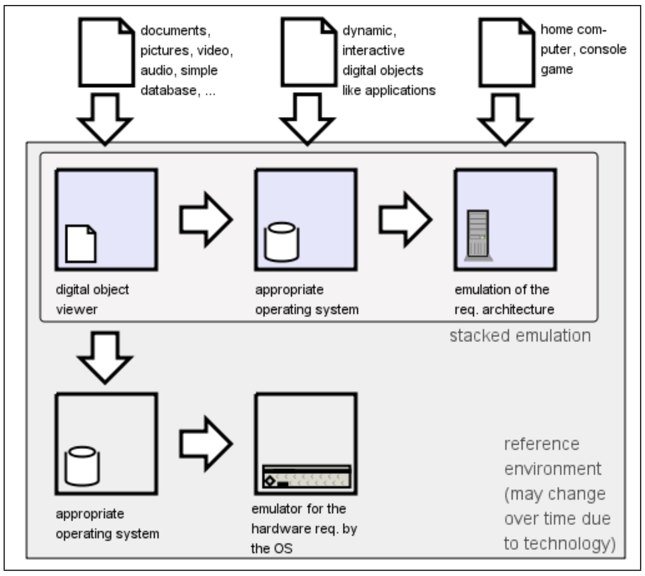

documents, pictures, video, audio, simple database, ...
dynamic, interactive digital objects like applications
home computer, console game
digital object viewer
appropriate operating system
emulation of the req. architecture
stacked emulation
appropriate operating system
emulator for the hardware req. by the OS
reference environment (may change over time due to technology)

Figure 3. Different view path depending on object type and archiving strategy

Step 4: Enabling Access to Emulation

In order to allow non-technical individuals to access deprecated user environments, the tasks of setting up and configuring an emulator, injecting and retrieving digital objects in and from the emulated environment have to be provided as easy-to-use services. Making these services web-based allows for a large and virtually global user base to access and work with emulated systems.

During the PLANETS project we developed the prototype GRATE[6] which allows the wrapping of various software environments within a single networked application. Designed as a general purpose remote access system to emulation services the architecture provides an abstract

[5] Alphaworks, *http://www.alphaworks.ibm.com/tech/uvc*

[6] GRATE – Global Remote Access To Emulation, *http://planets.ruf.uni-freiburg.de*

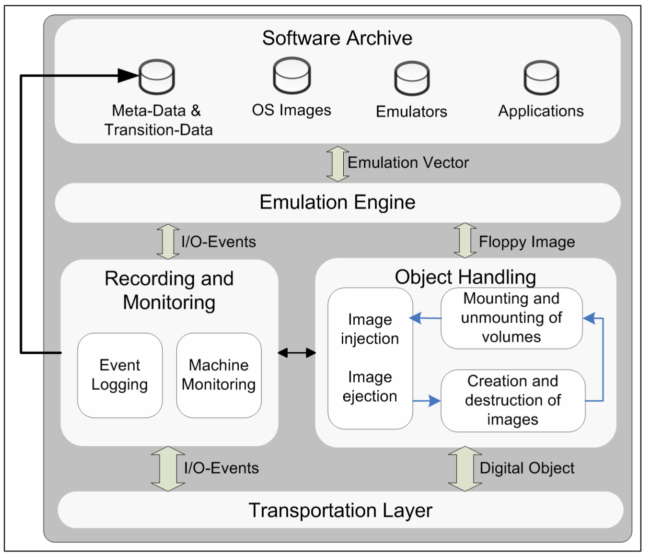


Figure 4. GRATE Architecture

interface independent of the digital object's type to users and thus was linked to other Web services like PLATO [2].

Screen output and input via mouse or keyboard – which until now are still the most used methods of human-computer interaction – are handled using an event and transportation layer. Currently events and screen output are transferred by using the open and widely used VNC protocol [9]. Figure 4 shows the general architecture of GRATE and its main building blocks. The access to digital objects does not depend on local reference workstations like in archives and libraries. By separating the emulation part from the archive user's environment, GRATE avoids a number of problems, like a sophisticated local installation of a range of software components in unpredictable user environments of different origins. The user does not need to be a trained specialist of ancient computer platforms, but in contrast it is equipped with an user interface similar to many Web 2.0 applications. Furthermore, this approach does not need to transfer proprietary software packages to end- user systems and thus might avoid licensing and digital rights management issues. The management of such services could be centralized and several institutions could share the workload or specialize on certain environments and share their expertise with others. Institutions like computer museums could profit as well, because they are able to present their collections in non-traditional ways rather than simply within their own room, consequently attracting more attention.

Another challenge arises from the transport of the requested artifact from the current into its original environment. Loading of digital objects is a major part of any automated processing setup. The file sets need to be passed into the emulated environment [15]. This is typically a non-trivial task and depends on the feature-set of the original environment. There are two challenges to be faced:

- Network transport from the user's site to the emulation Web service
- Local transport to the target environment

Emulators usually use special container files as virtual disk images. Therefore, they offer an option to transport a digital object into the emulated environment by embedding it in the container file, or by creating a secondary one, which is then attached as an additional virtual hard-disk. However, for producing or modifying such containers, exact knowledge of the internal format is required and usually additional tools are necessary. Furthermore, modifying container files usually cannot be done while the emulator is running, since changes to its internal structure might lead to a corrupt container file. In contrast, floppy and optical disks like CD or DVD are typically removable and thus offer a data exchange option while the emulator is running. Some emulators like QEMU support virtual media loading and ejecting functionality and media changes are noticed by the operating system. Not all hardware platforms and operating systems support optical drives, but most of them support floppy disks.

Step 5: Dealing with Interactivity

A further central problem next to the Framework integration lies in the automation of the human-computer-interaction. Typical digital objects were created with interactive applications on computer architectures with graphical user interfaces. The user was required to point and click using a pointer device (e.g. computer mouse) or using the keyboard to create or modify an object.

The traditional approach supporting the user to automate interactive tasks is the use of so-called macro-recorders. These are specialized tools to capture sequences of executed actions. However, this functionality is not standardized in terms of its usability and features. Not only are special software components needed, but knowledge on using such applications and operating systems is also necessary.

For a generic approach, a technical and organizational separation between the machine used for executing workflows and its input/output is required. Hence, emulated or virtualized environments are particularly well suited for recording an interactive workflow, such as installing a specific printer driver for PDF output, loading an old Word Perfect document in its original environment and converting it by printing into a PDF file. Such a recording can serve as the base for a deeper analysis and the generation of a machine script for the future than completely automated repetition. By using the aforementioned method, the authors demonstrated the feasibility of such simple migration task in an automated way [8].

An interactive workflow can be described as an ordered list of interactive events. Interactions might be mouse movements or keystrokes passed on to the emulated environment through a defined interface at a particular time. By using a generic approach to describe interactive events, there is usually no explicit feedback on executed interactive events. While a traditional macro-recorder has good knowledge about its runtime environment (e.g. is able to communicate with the operating system), in a generic emulation setup usually the screen output and the internal state of the emulated hardware are the only things visible (e.g. CPU state, memory). Furthermore, the recording/playback system has no knowledge of the system it operates. Hence a framework replaying a complete workflow in a reliable way is indispensable.

A solution relying solely on the time elapsed between recorded actions is not sufficient because executing recorded actions will take different amounts of time to complete depending on the load of the host-machine and the state of the runtime environment. Therefore, we link each interaction with a precondition and an expected outcome which can be observed as a state of the emulated environment. Until this effect is observed, the current event execution has not been completed successfully and the next event cannot be processed. While in the case of human operation the effect is observed through visual control in an automated run, an abstract definition of expected states and their reliable verification is necessary.

One suggested solution makes use of visual synchronization points [17]. For example, a snapshot of a small area around the mouse cursor can be captured before and after a mouse event and then used for comparison at replay time. Hence, replaying an interactive workflow becomes independent of computation time and the host-machine needs to complete a particular action execution. However, removing time constraints still does can not guarantee a reliable playback in general. First, if the synchronization snapshot is done in an automated way, important aspects of the observable feedback on executed actions might get lost. An optional manual selection of the snapshot area recording can improve the reliability since the user is carrying out the recording and is usually familiar with the interaction model of the graphical environment he operates. Second, mouse and keyboard events are passed on to the runtime environment through an abstract interface (e.g. through hardware emulation of a PS/2 mouse interface). Hence, sometimes the environment does not react to input events in the expected way. This occurs for example if the operating system is busy and unable to process input events. For reliable playback, such failures need to be detected and handled by the framework. Furthermore, the operator needs support to implement specific failure recovery strategies, e.g. resetting the machines to a stable previous state and retry the failed subsequence. Additionally, if the operator is able to attach meta data to specific events describing its original intend and possible side effects, not only will the reliability of automated execution be improved, but also specific knowledge on practical operation will be preserved.

To support these ideas, the interactive workflow has to be represented as a set of time-independent event transitions, relying only on valid and stable pre- and postconditions. For describing pre- and postconditions, the aforementioned visual snapshot technique was used, but extended to support users choosing the relevant snapshot

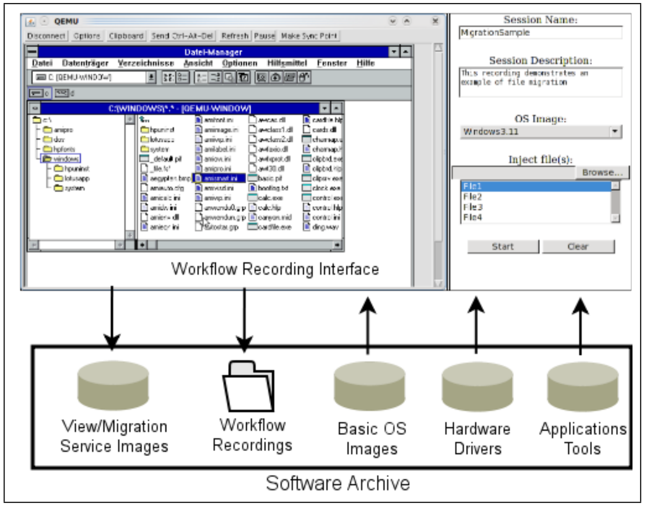

Session Name:
MigrationSample
Session Description:
This recording demonstrates an example of file migration
OS Image:
Windows3.11
Inject file(s):
File1
File2
File3
File4

Figure 5. Planets' Web frontend to emulation services needs to be extended with interfaces to software archive

area. The framework accepts three types of input events: keyboard entry, mouse events and special pseudo-events. Pseudo-events include specific control commands of the runtime environment (e.g. ctrl-alt-del) but might also be used to map the progress of longer running tasks (i.e. installation procedures) through empty dummy events. Mouse events cover pressing or releasing mouse-buttons and double-clicks. Especially since the abstract event passing interface provides no guarantees on action execution, a mouse pointer placement and verification system had to be implemented. Such a system not only makes mouse movement independent of the original users movements, but also allows users to jump to any previous event with a defined mouse pointer state. State transitions are triggered either through the arrival of appropriate feedback from the runtime environment or through a time-out. Failures can happen either by mismatching the precondition or the postcondition. If the precondition is not met within a defined time-out, the system may try to step back until a previous precondition matches and retry event execution from that point. In the case of a mismatched postcondition, the system could check if the precondition still holds and retry the last event execution. Although both recovery strategies may cover the most common failures, the operator still needs to decide which strategy is appropriate. The described approach is based on the GRATE system architecture using VNC for input/output abstraction.

Step 6: Preserving Necessary Software Components

A major factor in the discussion of emulation strategies is widely missing. The needed additional software components are implicitly used but are not categorized and officially archived. Thus a missing operating system or firmware ROM of a home computer might render a digital object completely unusable, even with a perfectly run-

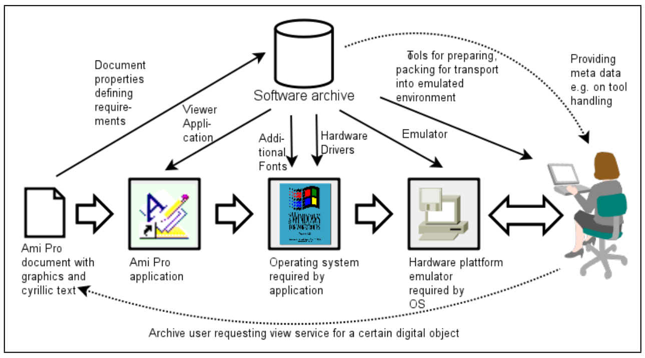

Software archive
Document properties defining require-ments
Viewer Appli-cation
Addi-tional Fonts
Hardware Drivers
Emulator
Tools for preparing, packing for transport into emulated environment
Providing meta data e.g. on tool handling
Ami Pro document with graphics and cyrillic text
Ami Pro application
Operating system required by application
Hardware plattform emulator required by OS
Archive user requesting view service for a certain digital object

Figure 6. Workflows and software components involved when accessing a digital artifact of a certain object type

ning virtual replacement of the original machine. A first step to formalizing the access to digital objects of different types were view paths (see Step 3). They do not only define workflows to be implemented as mentioned in the last section but generate lists of needed additional software components.

Rendering digital artifacts requires, depending on the object type, a large and complex set of software components, not only the original software application and operating system. Other dependencies such as font sets, decompression software, codecs for audio and video files, and hardware drivers for video output, sound cards and peripheral devices must be met as well (Fig. 1). Typically, the more recent the environment, the higher the level of complexity and number of different components required. In addition to storing and handling the digital objects themselves, it is essential that we store and manage this complex set of software components (Fig. 6). These dependencies and requirements can be formalised using view paths (pathways) both for emulation and migration approaches to preservation [16]. Despite the considerable efforts on digital preservation research, this essential groundwork has until now been largely neglected. This could lead to fatal gaps in the preservation workflows of future generations.

Another scenario where a comprehensive and well-managed software archive is essential is when a memory institution receives the legacy of an important writer, scientist or politician. Typically such archives have not been actively managed, but are nonetheless of importance for cultural heritage. Depending on the age of the material, software archeology techniques may be required to provide access to this material. Established preservation organisations such as libraries and technical museums would be the natural providers of such a capability.

Legal Issues Alongside managing the software components and associated documentation, a software archive must tackle the legal and technical problems of software licensing. A reputable institution must abide by the licences associated with the software it uses. For proprietary software, this may severely limit the rights of the institution to use the software to provide preservation services. Furthermore, technical approaches to protecting intellectual property, such as Digital Rights Management (DRM), copy protection mechanisms, online update or registration requirements all create significant problems for a software archive. To tackle this problem will require the cooperation of software manufacturers with a designated software archiving institution, to provide suitably licensed unprotected copies of software for long-term preservation purposes. We recommend the development of an approach similar in concept to the legal deposit approach used by many national or copyright libraries.

Step 7: Providing Reference Environments

The emulator has to be run in the environment the archive user is working with. The user is to be enabled to construct a view path from the original object. The base platform for emulation should be chosen from the most popular operating systems and computer hardware of a particular timespan. This prevents the establishment and costly operating of a hardware museum on one hand side and helps the user to orient him or herself more easily in a familiar surrounding. Additionally, the reference environment should offer easy access to all meta-data and required toolkits [16].

Every computer platform, historical as well as current, has its own complexities and concepts and most of the future computer users wont find old user interfaces as easy to use as we might think today. The same is true for set-up and installation routines of emulators and ancient operating systems. Another challenge arises from the transport of the requested artifact from the current into its original environment. Thus it would be desirable to automate the significant parts of the process in specialized departments of memory institutions with trained personnel and offer the services within a framework over the internet. This eases the complex procedures to be run on average computers and reduces the functionality to the viewer, e.g. in a web browser. The user gets the results presented via a virtual screen remotely on her or her computer (Fig. 4). With GRATE, a pilot was programmed to develop a prototype of an emulation service. This service is based on available open source emulators mentioned above and allows them to run on a remote basis.

Within the PLANETS framework [5], such emulation services can be integrated in more complex workflows of digital preservation. Emulated systems can be used as alternative endpoints of a migration workflow in order to allow an interactive view of the digital object in its original creation environment. Moreover, emulation itself could be used as a migration service in a different workflow. The PLANETS framework offers interfaces for web services for common tasks in digital preservation, like the characterization, validation, viewing, comparing, modifying and migrating of digital objects. Two PLANETS services are of particular interest for emulation.

The PLANETS *view* web services interface (Fig. 5) is designed to render a digital object. The service takes a digital object and returns a URI pointing to the rendered result. If the digital object requires a running rendering engine, the service offers methods for querying the engine's state and allows sending commands to it. The emulation viewing service offers access to already configured and ready-made emulators and software images. The web service accepts a list of digital objects and injects them into the running OS. The user is able to explore the environment, create, view or modify digital objects with their original application and compare the result visually with their appearance in current applications or migrated version of them.

By using the view interface for installing applications and their dependencies, not only can all steps of the recording procedure be recorded, but also might get annotated by the user. For each installation step this information is kept together with the system state before and after the installation, the application files and the system setup (e.g. which transportation option was used to provide the installation image) in the software archive. This way the software archive is able to:

- Calculate possible dependencies and view-paths for every known application setup;
- Ensure integrity of every view-path endpoint. This is achieved by keeping all intermediary setups and necessary installation files but most importantly the carried out installation steps;
- Calculate possible migration paths, provide access to the necessary files, the required set-up and the recording of the actual object migration.

The Planets Migrate Web service interface offers the ability to use various services to transform a digital object into a selected output format. The interface expects a digital object as input format and a designated output format accompanied with a list of service specific parameters. The outcome will be either a successfully transformed digital object or an error message. The *migration by emulation* services retrieves at instantiating time a so called view path-matrix from the software archive, which describes supported format migrations and then registers itself within the Planets framework. If the service is called with a supported view path, a view-path vector is requested from the software archive. This vector consists of a pointer to a system emulation engine, an appropriate runtime environment (e.g. a container file already set up with the appropriate operating system and applications) and a recorded interactive migration workflow. The digital object passed by the caller is injected into the runtime environment. After running the recorded workflow, the service returns all files (within a ZIP container) as digital object. Usually such a service is executed without visual control. However, for debugging and in case of an unrecoverable error, the view interface can be attached to the runtime environment.

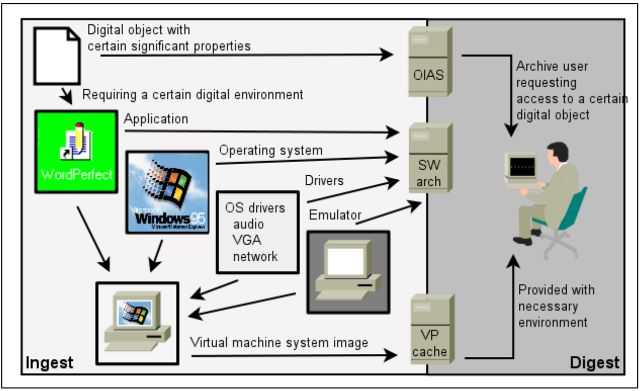


Figure 7. Future workflows to be implemented and integrated for emulation strategy

Conclusion and Future Work

Emulation is a very versatile and durable solution for retaining access to any kind of digital content. For some digital objects such as games, educational software or research applications, it is actually the only possible way as these objects usually can not be migrated. Nevertheless, emulation is not widely adapted to preservation frameworks and solutions in operation today for a number of reasons. There is still a trade-off between the well-established commercial virtualization tools without any long-term preservation focus like VMware and similar products, and preservation projects like Dioscuri and UVC are still missing major features to fullfill the average needs of memory institutions. UVC implements necessary web service interfaces for preservation frameworks but offers very limited support for different digital object types. Dioscuri still lacks the support of newer Windows and other operating systems from Windows 95 on.

Emulators like QEMU, MESS or ScummVM prove the validity of the Open Source approach. Especially QEMU has reached a stage rendering it suitable to be integrated into preservation strategies utilizing emulation. However, there is a gap in the development focus between the developer community of emulators on the one side and the professional deployment in long-term archiving on the other. The developers of the above-mentioned emulators have different development goals than archiving organizations. The present state of quality assurance is far from satisfactory and must be extended by suitable, preferably automatic, test scenarios which verify all important aspects of correct CPU and hardware replication. Generally, the question remains if the development methods followed by QEMU or Dioscuri, which originated from particular development environments and paradigms, will be valid for a long-term timespan.

Long-term perspective If one wants to ensure sustainability, the future development should be actively supported by a suitable syndicate, such as a large archiving organization. It shows that such a project can only be carried out with wide support from personal and Open Source Communities and needs a long-term perspective. A long-term archiving strategy like emulation can not be achieved by a single organization, even of the size of a national library, since the specific knowledge of particular computing architectures and digital object types will be spread between the archive and science communities. Projects like the Open Planets Foundation,[7] which continues the Planets archiving framework, could serve as an example for new approaches.

Nevertheless, one still needs to understand how to operate an old computer environment. Today, many of us still remember older environments such as MS-DOS and early Windows, but soon even those experiences will be lost. Thus an emulation strategy has to be supplemented with means to preserve a more complete idea of past digital environments rather than just their hardware emulators and software components. Therefore, manuals, tutorials and other supporting documents need to be preserved and kept available as well. Tacit knowledge could be preserved e.g. in workflow recordings of those past environments. This produces the base to offer appropiate access environments to the future archive users. Automated workflows will play a major role for migration-by-emulation strategies and the set-up of past digital environments for the deployment on reference workstations (Fig. 7).

Especially in regard to the preservation of a wide know-how, a distributed approach should be chosen in which single institutions specialize on one area but an intensive exchange and shared access to the repositories remains possible. Especially when it concerns the preservation of various localized variants of software, cooperation of the national institutions is proposed. A particular requirement of software archiving lies in the preservation of specific components, like hardware drivers for the network, graphics or sound cards offered by the emulators. In addition to this are codecs or fonts that are required for particular types of videos, audio or documents.

[7] OPF, *http://www.openplanetsfoundation.org*

1 REFERENCES

[1] David Bearman. Reality and chimeras in the preservation of electronic records. *D-Lib Magazine*, 5(4), 1999.

[2] Christoph Becker, Hannes Kulovits, Michael Kraxner, Riccardo Gottardi, Andreas Rauber, and Randolph Welte. Adding quality-awareness to evaluate migration web-services and remote emulation for digital preservation. In *Proceedings of the 13th European Conference on Digital Libraries (ECDL09)*, 2009.

[3] Max Burnet and Bob Supnik. Preserving computing's past: Restoration and simulation. *Digital Technical Journal*, 8(3):23–38, 1996.

[4] Adam Farquhar and Helen Hockx-Yu. Planets: Integrated services for digital preservation. *International Journal of Digital Curation*, 2(2), 2007.

[5] Ross King, Rainer Schmidt, Andrew N. Jackson, Carl Wilson, and Fabian Steeg. The planets interoperability framework. In *Proceedings of the 13th European Conference on Digital Libraries (ECDL09)*, pages 425–428, 2009.

[6] Raymond Lorie. *The UVC: a Method for Preserving Digital Documents - Proof of Concept*. IBM Netherlands, Amsterdam, PO Box 90407, 2509 LK The Hague, The Netherlands, 2002.

[7] Commission on Preservation, Access, and The Research Libraries Group. Report of the taskforce on archiving of digital information. WWW document, *http://www.clir.org/pubs/reports/pub63watersgarrett.-pdf*, 1996.

[8] Klaus Rechert and Dirk von Suchodoletz. Tackling the problem of complex interaction processes in emulation and migration strategies. *ERCIM News*, (80):22–23, 2010.

[9] Tristan Richardson. The rfb protocol. WWW document, *http://www.realvnc.com/docs/rfbproto.pdf*, 2009.

[10] Jeff Rothenberg. Ensuring the longevity of digital information. *Scientific American*, 272(1):42–47, 1995.

[11] Jeffrey van der Hoeven. Dioscuri: emulator for digital preservation. *D-Lib Magazine*, 13(11/12), 2007.

[12] J.R. van der Hoeven, R.J. van Diessen, and K. van der Meer. Development of a universal virtual computer (uvc) for long-term preservation of digital objects. *Journal of Information Science*, 31(3):196–208, 2005.

[13] Raymond van Diessen and Johan F. Steenbakkers. *The Long-Term Preservation Study of the DNEP project - an overview of the results*. IBM Netherlands, Amsterdam, PO Box 90407, 2509 LK The Hague, The Netherlands, 2002.

[14] Remco Verdegem and Jeffrey van der Hoeven. Emulation: To be or not to be. In *IS&T Conference on Archiving 2006, Ottawa, Canada, May 23-26*, pages 55–60, 2006.

[15] Dirk von Suchodoletz. *Funktionale Langzeitarchivierung digitaler Objekte – Erfolgsbedingungen für den Einsatz von Emulationsstrategien*. Cuvillier Verlag Göttingen, 2009.

[16] Dirk von Suchodoletz and Jeffrey van der Hoeven. Emulation: From digital artefact to remotely rendered environments. *International Journal of Digital Curation*, 4, 2009.

[17] Nickolai Zeldovich and Ramesh Chandra. Interactive performance measurement with vncplay. In *ATEC '05: Proceedings of the annual conference on USENIX Annual Technical Conference*, pages 54–64, Berkeley, CA, USA, 2005. USENIX Association.