Challenges for the Evaluation of Emulation

Mark Guttenbrunner

Vienna University of Technology, Vienna, Austria
http://www.ifs.tuwien.ac.at/dp
guttenbrunner@ifs.tuwien.ac.at

Abstract. Emulation is one of the main strategies for preserving digital objects. With different emulation environments to choose from when preserving a collection of digital objects it is necessary to choose the emulator that is able to render the significant properties of an object best. But also on the development side it is necessary to run automated tests for new versions of emulators.
In this work we present some of the main challenges for automated testing of emulators with specific digital objects. To compare different original and emulated environments it is necessary to document the environment used to render an object. Changes in this 'view-path' comprising all secondary digital objects needed to perform the emulation can lead to changes in behavior or appearance. For interactive and dynamic objects it is necessary to determine the external events influencing an object, e.g. user input and changes in synchronized timing can lead to different results when executing a digital object.
Significant properties of descriptive forms, i.e. the physical manifestations of an object, are compared when using migration as a strategy by comparing the original and the migrated form. With emulation the descriptive form of the object remains unchanged, so properties of a rendered version of the object have to be compared to determine if the preservation was successful. Emulation environments have to be extended to allow for the extraction of significant properties of digital objects. These properties can vary over different points in the rendering process.

1 Introduction

Research in preservation planning and evaluation of tools and objects has been made mainly for migration as a digital preservation strategy. But migration is not always a suitable strategy. For dynamic and especially interactive digital objects emulation is an important strategy as well. It is well known how to extract and compare significant properties for migrated objects, but with emulation the original object is unchanged. Instead of comparing an original and a migrated version of a digital object the comparison of a rendered version of the object in its original and in an emulated environment is necessary to determine if the significant properties of the object stay intact.

Similar to the migration of digital objects the goal of the evaluation of emulation environments is to perform repeatable experiments that allow us to take an informed and accountable decision on the best emulation environment for a certain digital object. To achieve this we have to extract significant properties from the rendering process. Automatic comparison of significant properties extracted from different emulation environments should be made possible. These steps would allow us to do preservation planning for emulation environments and automate parts of the process of testing emulators.

This article shows the challenges of evaluating emulation environments. It is structured as follows. First an overview of related work is given in Section 2. Then the importance of rendering properties for different types of digital objects for which emulation could be a suitable preservation strategy is shown. Next we discuss the necessity of documenting the environment and automating external events. We take a look at the rendering process and the extraction of significant properties from the emulation environment. Finally in Section 8 we present the conclusions and discuss what future work has to be done.

2 Related Work

Migration ([10]) and Emulation ([7], [12]) are listed in the UNESCO guidelines for the preservation of digital heritage [13] as the main strategies for digital preservation. Emulation refers to the capability of a device or software to replicate the behavior of a different device or software. It is possible to use hardware to emulate hardware, software to emulate software or software to emulate hardware. The challenges presented in this article mainly use *Emulator* as defined in [8] for a program that virtually recreates a different system than the one it is running on, but most of the presented work applies to Emulation in a broader context as well.

Previous research has been done on methods for evaluating the effects of migration on documents. A preservation planning workflow is described in [9] and allows for repeatable evaluation of preservation alternatives. An implementation of this workflow has been done in the preservation planning tool *Plato* ([1]), utilizing automatic characterization of migrated objects with tools like *Droid* ([3]) to identify files. The significant properties of migrated objects can be compared automatically using the *eXtensible Characterisation Language* (XCL) ([2]) to measure the effects of migration on the object. While the preservation planning tool can be used to compare emulation environments as shown in a case study in [5], the comparison has to be done manually. Significant properties of software as one category of dynamic objects are listed in [6]. In [11] the information contained within a file is distinguished from the rendering of this information. To compare emulation environments we have to compare information about the rendered object as the object itself is unchanged in different environments.

3 Rendering Properties for Object Types

In [4] the following types of interactive digital objects are described as candidates for using emulation as a digital preservation strategy: application software, dynamic documents, interactive art and video games. For these object types emulation is an obvious choice to preserve the interaction properties. But all other non-interactive digital objects can be candidates for preservation through emulation as well. If objects have to be kept in the original format (e.g. for legal reasons) even for static documents emulation might be the strategy of choice.

Especially for dynamic objects properties of the rendering process that are not encoded in the descriptive form, i.e. the physical manifestation can be relevant as well (e.g. frame rate, CPU-cycles in a certain amount of time, number of disk operations, reaction time between user input and resulting change of the rendered object). These significant properties have to be determined for an object or a class of objects that have to be preserved.

4 Documentation of Environment

One of the steps in the preservation planning workflow defined in [9] is to describe the conditions under which a migration of objects was performed. For emulation this is a crucial step as it is necessary to define the environment in which the object is executed in. Every setting of the rendering system can influence the behavior and appearance of the object. Besides the settings for hardware, operating system and the digital object itself, other digital objects influencing the objects rendering process (e.g. additional software on a system influencing the speed, operating system plug-ins effecting the appearance) have to be considered as well.

For every digital object a view-path of necessary secondary objects can be constructed. Secondary digital objects are software that is needed for rendering the object that has to be preserved. As an example an operating system and a viewer application might be the minimal view-path to render an image. As the same image can be rendered using different viewer applications not necessarily running on the same operating system, results of the rendering process can be different (Figure 1). Depending on the object environment settings can influence behavior and appearance. While video games usually have an appearance that is independent from settings, the look and feel of application software could be changed in the operating system configuration.

To minimize the side effects of changes in environment on rendering, the view-path with all relevant settings should be well documented. This lays a foundation for evaluating emulators. By keeping the view-path constant we can make sure that differences in rendering are caused by differences in emulation environments and not by changes in the view-path.

5 Automating External Events

Depending on the object type different external events influence the behavior or the appearance of an object. Changes in these events between the original environment and an emulated environment can change the behavior of an object. Three major influences on dynamic objects are listed below.

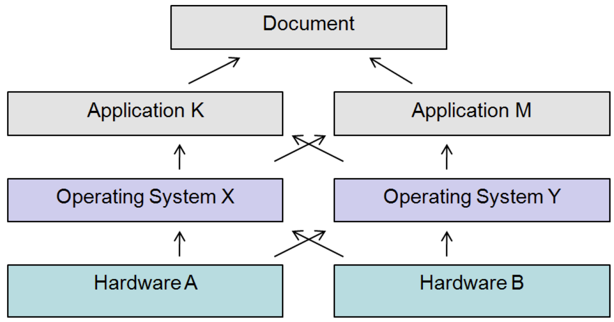

Fig. 1. Different view-paths for rendering a digital object.

5.1 User Input

For interactive objects changes in user input influence the behavior of an object. To compare the behavior of an object in the original and the emulated environment for testing purposes we therefore have to unify not only the inputs that were triggered by a user but also the precise moment at which those inputs occurred. Depending on the object the timing might be more or less crucial. A mouse click that is 1 second to late might not make a difference for a database application, while it completely might change the events in a video game or a piece of interactive art.

As it is impossible to manually apply the same interaction twice at the precise same moment during the emulation process, it is necessary to automate the application of user input. This is currently either not supported by emulators or supported in proprietary format. Input scripts recorded on one environment (original or emulated) cannot be applied to other emulation environments.

5.2 Synchronized Timing

Behavior of objects can be bound to time variables. These are usually hardware specific registers that can be read using software. Especially for early video games the timing of graphic objects is often bound to the location of the raster beam on the screen. Differences in emulation of the timing and the original timing can lead to unwanted differences in the displayed image. But not only the raster beam, also processor interrupts or hardware system clocks are potential causes of changes in the rendering process for time critical code.

To avoid side effects due to differences in timing it is necessary to synchronize timing to the start of the emulation process, e.g. a hardware clock has to be set to the same value in different emulation environments to synchronize the events instead of using the hardware clock of the host system.

5.3 Randomization

Especially video games and interactive art objects use random events. The occurrence of random events is usually tied to a random number generation. The algorithm that generates the number has to be initialized by setting a base number that is set using an indeterminate hardware number (e.g. hardware clock, raster beam, timed user input).

By identifying the source for the random number based algorithm and keeping the generation of the number constant over different emulation environments it is possible to eliminate the randomness of the algorithm.

6 Rendering

Significant properties of an original and a migrated version of the descriptive form of an object are usually compared when using migration as a digital preservation strategy. If the migrated version has the same significant properties as the original version, then the tool used to migrate the object is considered a successful preservation action. For emulated environments as preservation action tools the situation is different. The descriptive form of the object is unchanged for the original environment and different emulated environments. Rendered version of the object have to be compared instead of descriptive versions.

In an environment used to render the object usually various different forms of the object exist. It could be stored and rendered in a location in system memory after being loaded by a viewer application, stored in specific memory of the outputting hardware (e.g. video card) or output on the output interface of the system (e.g. display device). To compare an object in different environments the same rendered version of the object has to be used and extracted from the environment.

7 Extraction of Significant Properties

Rendering of a dynamic object is a continuous process. The significant properties of an object can change likewise during the rendering process in different states. Depending on the object some of these states might be significant while others might not be. A static document will have only one significant state once it is loaded and displayed while loading a series of web-pages creates various significant states when the different web-pages are displayed. In the case of a video game every rendered frame on the screen can be significant to preserve the changing image. Extracting properties therefore can also be done at one moment of emulation, at various states during the execution of an object or as a continuous stream.

With migration as a strategy it is possible to create tools that examine the migrated object and extract significant properties. For emulation the properties have to be extracted from the rendering process. As not all information about the rendering process is visible on the rendered object it is necessary that the environment supports the extraction of defined properties.

8 Conclusions and Future Work

In this paper we discussed the challenges and requirements for the evaluation of emulation environments in the context of digital preservation and for selection of the best emulation tool for a set of digital objects.

Depending on the object type not only the properties of an object stored in the descriptive form, but also properties of the rendering process can be relevant. By documenting the reference environment in which the object is originally executed along with all secondary objects potentially influencing its rendering we can minimize side effects on appearance and behavior due to changes in the environment. The significant properties of the object have to be defined and random elements have to be made deterministic to create constant behavior and appearance. Decisions have to be made on what level to compare the rendered object and in what regularity (once, at specific times or continuously).

Future work should be done on the development of guidelines for the support of automated user interaction in emulators. Significant properties that can be extracted from the emulation environment have to be defined and mapped to a characterization language like XCL. Emulators should include the possibility to extract information in the defined format as well as support for the extraction of rendered forms of the object at specified points of emulation.

Acknowledgements

Part of this work was supported by the European Union in the 6th Framework Program, IST, through the PLANETS project, contract 033789.

References

1. Becker, C., Kulovits, H., Rauber, A., and Hofman, H. Plato: a service-oriented decision support system for preservation planning. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries (JCDL'08)* (June 2008), ACM.
2. Becker, C., Rauber, A., Heydegger, V., Schnasse, J., and Thaller, M. Systematic characterisation of objects in digital preservation: The extensible characterisation languages. *Journal of Universal Computer Science 14*, 18 (2008), 2936–2952. `http://www.jucs.org/jucs_14_18/systematic_characterisation_of_objects`.
3. Brown, A. Automatic format identification using pronom and droid. *Digital Preservation Technical Paper 1* (2008). `http://www.nationalarchives.gov.uk/aboutapps/fileformat/pdf/automatic_format_identification.pdf`.
4. Guttenbrunner, M. Preserving interactive content: Strategies, significant properties and automatic testing. In *Workshop on Data Analysis (WDA'2008)* (Dedinky, Slovakia, June 2008).
5. Guttenbrunner, M., Becker, C., Rauber, A., and Kehrberg, C. Evaluating strategies for the preservation of console video games. In *Proceedings of the Fifth international Conference on Preservation of Digital Objects (iPRES 2008)* (London, UK, September 2008), pp. 115–121.

6. Matthews, B., McIlwrath, B., Giaretta, D., and Conway, E. The significant properties of software: A study. JISC Study, 2008. http://www.jisc.ac.uk/media/documents/programmes/preservation/spsoftware_report_redacted.pdf.
7. Rothenberg, J. *Avoiding Technological Quicksand: Finding a Viable Technical Foundation for Digital Preservation*. Council on Library and Information Resources, January 1999. http://www.clir.org/pubs/reports/rothenberg/contents.html.
8. Slats, J. Emulation: Context and current status. Tech. Rep., 2003. http://www.digitaleduurzaamheid.nl/bibliotheek/docs/white_paper_emulatie_EN.pdf.
9. Strodl, S., Becker, C., Neumayer, R., and Rauber, A. How to choose a digital preservation strategy: Evaluating a preservation planning procedure. In *Proceedings of the 7th ACM IEEE Joint Conference on Digital Libraries (JCDL'07)* (June 2007), pp. 29–38.
10. Testbed, D. P. Migration: Context and current status. White paper, National Archives and Ministry of the Interior and Kingdom Relations, 2001.
11. Thaller, M. Interaction testing benchmark deliverable PC/2 - D6. *Internal Deliverable, EU Project Planets* (2008).
12. van der Hoeven, J., Lohman, B., and Verdegem, R. Emulation for digital preservation in practice: The results. *International Journal of Digital Curation 2*, 2 (2007), 123–132.
13. Webb, C. *Guidelines for the Preservation of the Digital Heritage*. Information Society Division United Nations Educational, Scientific and Cultural Organization (UNESCO) – National Library of Australia, 2005. http://unesdoc.unesco.org/images/0013/001300/130071e.pdf.