SOFTWARE PRESERVATION IN THE NETHERLANDS: LOWERING THE THRESHOLD FOR CULTURAL HERITAGE INSTITUTIONS

Röck, C.; de Vos, Jesse; O'Donohoe, Eoin

Publication date
2021

Published in
iPRES 2021 Proceedings

Software Preservation in the Netherlands

Lowering the Threshold for Cultural Heritage Institutions

Eoin O'Donohoe
Sound and Vision
Netherlands
eodonohoe@beeldengeluid.nl
ecodonohoe@gmail.com

Claudia Röck
Sound and Vision
Netherlands
crock@beeldengeluid.nl
claudia.roeck@crockodile.ch
0000-0003-2612-5672

Jesse de Vos
Sound and Vision
Netherlands
jdvos@beeldengeluid.nl
0000-0003-0583-1877

Abstract – Thus far, software preservation in Dutch Heritage institutions is far from common practice. Over the course of 18 months the collaborative Software Archiving-project aimed to lower the threshold for institutions to start or advance software preservation. The project investigated the status quo, made recommendations and offered practical resources on emulation as a service, browser emulation and metadata for software preservation.

Keywords – Emulation, Software, Web browsers, Metadata

Conference Topics – Latest developments in tools, strategies, and practices, in preservation of research data, software, social media, web content, rich or interactive or smart media, VR/AR, etc.

I. Introduction

The complexities of software preservation are manifold. For a large part these are universal problems of obsolescent technologies, software dependencies, restrictive legal frameworks and complexities around definitions and ontologies that need to be addressed in metadata. Because of their universal nature, software preservation lends itself especially well to a collaborative approach. In 2019 the Dutch Digital Heritage Network (NDE) therefore made funding available for a project on the preservation of software with as its goal to lower the threshold for cultural heritage institutions (CHIs) within the network to start or advance their software preservation activities. In this paper we will report on some of the findings of that project. First we performed a number of interviews to identify the challenges institutions were facing in starting or advancing software preservation. The remainder of the 18-month project we focused on testing the EaaS framework as a means of providing access to software archives, browser emulation as a specific implementation of software preservation and metadata for describing software in archives.

II. Software Preservation in the Netherlands

Initially the project would perform a survey among CHIs in the Netherlands that have software in their collections. However, it turned out that the number of institutions having such collections were few and far between, too few in fact to render any quantitative analysis of the current status quo useful. Therefore a number of semi-structured interviews were performed with representatives of archives that held software in their collections.

On a policy level software archiving was found to be quite underrepresented. Dutch CHIs that engage with software as archival objects do so in a more experimental manner. They largely distinguish between two motivations for the preservation of software. One is where software itself is considered an object of cultural-historical value in and of itself, e.g. a computer game or a software-artwork. The other reason to preserve software is because it is

seen as a precondition for access to files that exist elsewhere in the collection, e.g. a CAD-file containing an architectural design that needs a specific version of AutoCAD to be opened. The people interviewed all agreed that the second scenario described already is unavoidable for more complex file formats that have specific dependencies. This is a challenge to most institutions whose preservation policies are focused on the bulk work of standard file formats and file format migration. Considering software itself as cultural objects will require changes to collection policies.

Based on these interviews a number of recommendations were made to the network of CHIs. A collaborative research agenda on software preservation could be a way to work together to tackle the challenges software preservation presents us with. Also, the possibilities of a shared emulation infrastructure for access is worth exploring. The lessons learned on an international level (in the EaaSI project[1] especially) can be used to advance such collaboration in the Netherlands. Finally, investing in developing better technological skills among archivists is absolutely essential if we're to adopt the technologies and practices needed for the preservation of software.

III. Emulation as a Service

Emulation as a Service (EaaS) endeavours to provide long-term preservation and access to digital material through emulation. A key goal is to simplify the process and management of emulation components. The framework has been in development by the bwFLA team at the University of Freiburg since 2011 and now operates under the OpenSLX label. It makes use of abstract emulation components to standardize deployment and to hide individual emulator complexity. [1] The user of the framework does not need to worry about the full technical workings of the emulator in use as the framework is designed to interact with the prepackaged, or containerised, emulators provided. These containers can be slotted in easily via the framework's front end without the need for users to understand what is going on in the background. The framework supports all major desktop systems and utilises a host of containerised, open source emulators including Qemu, Basilisk II, Sheepshaver and Vice.

In order to test the possibilities and limitations of the EaaS framework we selected several use cases, suggested by our project partner institutions, that covered a broad range of platforms, software and file types. These included a Commodore 64 games collection from the Regionaal Archief Alkmaar, a batch of old QuarkXpress design files from the Het Nieuwe Instituut in Rotterdam, dating from between 1995 and 2005, and, from Beeld en Geluid's own collection, a selection of PC games from the late nineties. At all stages of research and testing we were influenced and guided by the Emulation as a Service Infrastructure (EaaSI) project, led by Yale University Library, which has been working towards the development of technology and services to expand and scale the capabilities of the EaaS software since 2018. Much of their work has revolved around the establishment of a network of partner institutions to share the testing, research, and improvements of the framework.

The key to using the EaaS framework depends on the setting up of appropriate environments in which a user can interact with a specific file or piece of software. The basic principles of Objects, Software and Environments are what make up an EaaS session. An Object can be anything from a piece of software or game to a virtual disk or datafile. Objects form the first layer of the environment creation process and can be imported into the framework using the designated page. This could be the image of an installation CD or an archive file packaging software components or a collection of specific file formats. Objects can be promoted to Software in order to keep separation between software applications and other datafile objects. The environment creation process relies on multiple components, ranging from operating systems, commercial and open source software applications, device drivers, etc., in order to work. By promoting these Objects to Software it is possible to keep a clearer separation between the components that comprise the emulated

[1] https://www.softwarepreservationnetwork.org/emulation-as-a-service-infrastructure/

environments and the files which are themselves the target of emulation. Environments rely on a combination of a suitable emulator, to replicate the hardware, and an Operating System, in the form of Software, upon which further Software can be installed. The EaaS framework provides the option for multiple configurations of such Environments to be saved for future use.

For the purposes of this study, individual environments and their respective components were researched, sourced and configured in order to test each use case. This included investigating and documenting the specific requirements for each component of the software/hardware stack. With a specific file type or software version as a starting point it is possible to consult available documentation to determine compatibility with operating systems and hardware in order to piece together a suitable Environment. This has the potential to be a time consuming process but, as detailed by the continuing work of the EaaSI project, such Environments can be shared and made accessible across multiple institutions thus spreading the workload and helping build towards a common infrastructure.

For each of the use cases a feedback workflow and survey was designed in order to evaluate and gain feedback on the key operations of using the framework and also the quality and accuracy of the emulated file. The results of the performance and integrity of the emulated files varied upon the intensity of the session. Noticeably, the more graphically intensive examples, such as video games, suffered from more lag when rendering when compared to desktop publishing software. Overall, the testing carried out demonstrated that emulation through the EaaS framework is a viable option. The sheer variety of collections and workflows within heritage institutions is sure to present both opportunities and challenges going forward, but the ongoing development and support of the software by OpenSLX, and the extensive implementation work carried out by the EaaSI team provides a valuable launchpad for getting started.

IV. Browser Emulation and Characterisation

One of the use case partners in the project is LIMA, a member of the NDE network and an organisation for the collection, research, and preservation of media art in Amsterdam. They host netart on a web server specifically set up for this purpose. As these artworks are a few years old LIMA wondered what web browsers should be used to present them. Not every web browser renders a website in the same way and therefore browsers offer an interesting use case where software preservation meets web preservation. Web browser developers and other sources such as Wikipedia can provide information about the technical properties of web browsers, they provide only very limited descriptions of browser behaviour. This is why this part of the project investigated how web browsers could be conceptualised in terms of website authenticity and in regard to the browsers' version, and feature history.

Browsers opened the doors to web culture making interactive websites with embedded media accessible to a wide public. The first browser war[2] between Netscape Navigator and Internet Explorer, beginning in the mid-1990s and ending in the early 2000s was the driver for the development of many new advancements in browser technology. Many websites used browser-specific features that could not be interpreted by all browsers of the time. The second browser war[3] between 2004 and 2017 saw new browsers such as Google Chrome, Opera, and Firefox14 emerge and the dominance of Internet Explorer decrease. In contrast to the first browser war, this period was dedicated to the expansion of web standards—, therefore, the differences between web browsers of this era are less pronounced than during the first browser war.

One important feature of web browsers is their ability to present multimedia content. From 1995 until about 2018, interactive animations and graphics in websites were enabled by browser plugins such as Java and Flash. They were enthusiastically used by companies, artists, and game developers and are deprecated today. With the advent of HTML 5 and the WebGL-API and the

[2] https://en.wikipedia.org/wiki/Browser_wars accessed 2020/07/30

[3] https://en.wikipedia.org/wiki/Browser_wars accessed 2020/07/30

evolution of Javascript, browsers are no longer dependent on external web plug-ins.

Besides a technical description of web browsers (such as compatibility with plugins, ability to render unencrypted websites), web browsers and their versions can be characterised according to their behaviour:

- Positioning and scaling of website elements
- Look, sound, and movement of web animation
- Reaction to key or mouse input
- Fonts used
- Look of browser and browser elements (scroll bar, mouse pointer, pop-up windows)
- Browser specific add-ons or toolbars
- Browser specific (non-standard) HTML commands
- Bugs in the browser engine (bugs can be exploited by web designers and artists)
- Colours. Certain old browser versions cannot interpret colour profiles correctly.[4]

However, web browsers and artifacts caused by web browsers are not described systematically. For audiovisual media, there is a website that gathers their typical artefacts and properties: http://www.avartifactatlas.com/. This could serve as an example for setting up a similar collection of web browser characteristics. Such a collection would describe the web browser environment and a website that can reproduce a specific browser artifact. Ideally, the browser environments would be made available in emulation as a service. Oldweb.today[5] is a service provided by Rhizome giving access to a given selection of web browsers. It is possible to render live web pages and web archives with it. But the access is limited to specific browsers and the suggested database with browser characteristics does not exist yet. Due to the huge number of web browser environments, this could only be realised in a collaborative effort. It would help to choose the right web browser environment for a specific website or a collection of websites.

The expression "browser emulation" is a common expression in digital art preservation, although it is not technically accurate: it is not in fact the browser that is being emulated, but the hard- and software environments that are running it. Specific web browser versions only work on corresponding computer hardware, so old hardware has to be emulated in order to run obsolete browsers. A browser emulation can be set up in a way that it is automatically launched when a user accesses the obsolete website[6]. It can resolve access problems due to obsolete plug-ins, unencrypted websites, and obsolete browser features. It cannot fix problems server-side, such as outdated script language and database versions, or provide access to external data that has been dislocated, deleted, or otherwise changed.

The tests with browser emulation showed that not all the web browsers versions and their plugins are made available by their software companies[7]. As a preventive measure the preferred browser environment and its components should be acquired together with a web archive or a website.

With this investigation we hope to raise awareness for the specific needs of obsolete websites and the variety of web browser characteristics. At the same time this research served to test the emulation as a service infrastructure. Sharing of created environments is one of the biggest benefits of EaaS and would greatly facilitate the display of obsolete websites.

V. Outlook

The project is being finalised at the time of submission of this paper. A report on how to structure metadata in order to ensure future access to archived software is close to completion. In it we investigate how to leverage PREMIS and Wikidata to describe software in a standardised and, where possible, machine readable way. We are hoping that the combination of the deliverables mentioned and

4 https://developer.mozilla.org/en-US/docs/Mozilla/Firefox/Releases/3.5/ICC_color_correction_in_Firefox and https://cameratico.com/color-management/firefox/ accessed May 2021

5 https://oldweb.today/ accessed May 2021 [2]

6 Rhizome is using browser emulation (remote browser) for the presentation of several netart pieces. [3]

7 Google does not provide access to old Google Chrome versions.

sharing our positive experiences with the EaaS framework and providing guidelines and recommendations motivates Dutch archives to take the next step with software preservation.

ACKNOWLEDGMENT

This project was made possible with funding of the Dutch Heritage Network (NDE) and participation of our project partners, Regionaal Archief Alkmaar, Het Nieuwe Instituut, The Netherlands Institute of Sound and Vision, LIMA and the Netherlands eScience Center.

REFERENCES

[1] Cochrane E, Rechert K, Anderson S, Meyerson J, Gates E (2019) Towards a universal virtual interactor (UVI) for digital objects. In: iPres2019 (ed): Conference Proceedings, Amsterdam

[2] Connor M, Dean A, Connor MJ, Espenschied D (eds) (2019) The art happens here: Net art anthology. RHIZOME, New York, p. 436 ff

[3] Boss K, Steeves V, Rampin R, Chirigati F, Hoffman B (2019) Saving Data Journalism: Using ReproZip-Web to Capture Dynamic Websites for Future Reuse. In: iPres2019 (ed): Conference Proceedings, Amsterdam