Oldweb.today: Browsing the Past Web with Browsers from the Past

Dragan Espenschied and Ilya Kreymer

Abstract Webpages have long stopped being just static "documents". Since the introduction of inline graphics and JavaScript, they have moved towards becoming executable code dependent on specific browsers and feature sets. On the basis of examples from the history of net art and the legacy browser service oldweb.today, this chapter presents the value of preserving browser software along with web archives.

1 The Longevity of Web Archives

Webpages have long stopped being just static "documents". Since the introduction of JavaScript with Netscape version 2 in 1995, webpages have increasingly developed towards becoming executable code that is dependent on the right software environment—the browser—not only to "render" correctly but to "perform" correctly. Only when combined with the right browser from the past will a webpage from the past appear as it used to.

However, so far, the established practice of web archiving is mainly concerned with static resources, such as HTML pages, JPEG images, and so on, which are first captured from the live Web and then stored in a collection to be accessed later.

As for other digital preservation practices, the storage format specifically developed for web archiving, WARC,[1] has been designed to abstract certain complicated

[1]WARC is standardised by ISO; the specification can be found on the International Internet Preservation Consortium's GitHub at https://iipc.github.io/warc-specifications/specifications/warc-format/warc-1.1/

D. Espenschied (✉)
Rhizome at the New Museum, New York, USA
e-mail: dragan.espenschied@rhizome.org

I. Kreymer
Lead Developer Webrecorder and oldweb.today, San Francisco, USA

D. Gomes et al. (eds.), *The Past Web*,
https://doi.org/10.1007/978-3-030-63291-5_20

technical and organisational issues to increase a collection's usefulness as far as possible into the future. Microsoft Word documents are often converted to PDF for archival purposes so as not to have to deal with a complicated legacy Office software stack in the future. Similarly, web archiving is not about copying a remote web server, including all of its software such as databases, scripting languages, or any other components that might be involved in producing a live website. The WARC format describes the requests and responses exchanged with web hosts. Essentially, it preserves information at the level of HTTP, the standardised communication protocol that underlies the Web.

When accessing a web archive, a user requests content referenced by URL at a point in time, and a "Wayback Machine" or similar mechanism selects the closest matching resources from a collection's WARC files and sends them to the user's browser. As long as the WARC files are carefully stored and the Wayback Machine[2] works, the archived websites should stay available in perpetuity. However, looking back on more than 25 years of web archiving shows that this is not true. The farther away in the past websites were captured, the higher the likelihood of their looking odd, missing critical elements or behaving differently, even if all resources are still stored in the WARC files. This effect is most obvious with now deprecated browser plug-ins like Flash or discontinued integrations like Microsoft's ActiveX but can also appear in other areas. While this feels like some force of nature is at work and data is "rotting" or "degrading" in the archive, the data is not changing at all—it remains immutable. What is changing is the software on the user's end, since the final stage of assembling and performing an archived webpage is not handled by the Wayback Machine but by the user's browser. Even when considering very recent history, a website captured in 2015 was probably created with Chrome version 42 in mind; in 2020, it might be accessed with Firefox version 72. A lot of things changed in browsers within just 5 years: JavaScript engines were stripped of some functions, autoplay of audio and video was disabled by default, and new options such as blocking social media widgets or adopting "dark mode" became available to users, allowing them to influence how a website looks and behaves. At some point, the sum of changes in browsers over time will inevitably affect how sites captured in the past appear at the time of access. While the above example of two mainstream browsers released 5 years apart already highlights considerable differences, looking at a 2003 website optimised for Internet Explorer 6 in the latest Chrome browser two decades later (see Fig. 1) results in a heavily distorted view: an empty space filled with clouds divides the landscape like a ravine, with a palm tree hovering above. Users have no way of knowing or reasoning that a Java applet[3] is supposed

[2]A software system providing archived web resources to users is typically called a "Wayback Machine", after the first of its kind made accessible by the Internet Archive in 2001. In the meantime, additional tools have been created to "replay" web archives, such as the open source fork of the original Wayback Machine, OpenWayback, or new open-source projects like pywb and wabac.js that originated from the Webrecorder project.

[3]Java applets were small, platform-independent programs to be executed in the browser via an embedded Java virtual machine and initially designed for rich interactions beyond the Web's

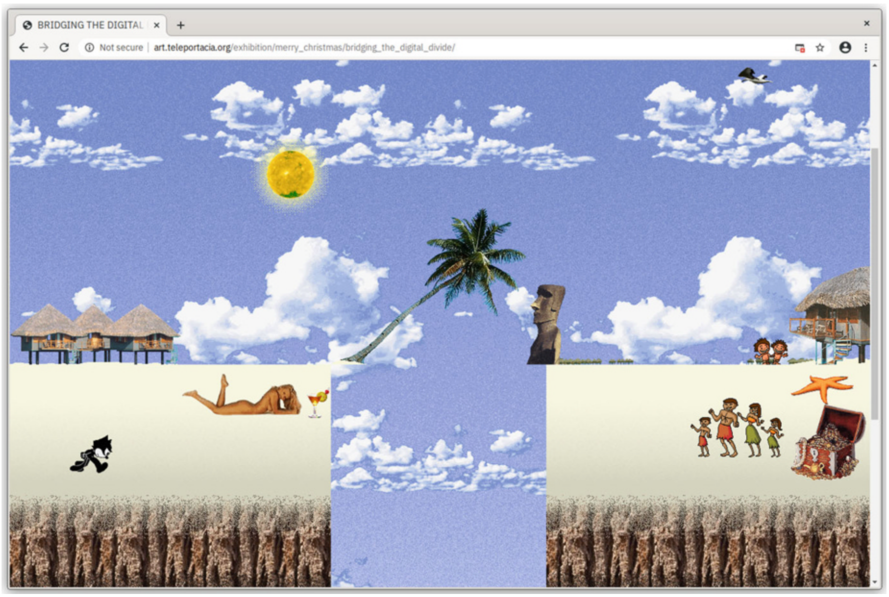

BRIDGING THE DIGITAL
Not secure
art.teleportacia.org/exhibition/merry_christmas/bridging_the_digital_divide/

Fig. 1 Dragan Espenschied, *Bridging The Digital Divide*, 2003, as shown in Google Chrome version 80 on Linux in 2020

to be displayed in this gap. The browser simply ignores the applet without giving any notice in the user interface.

2 Characteristics of Browsers

The history of web browsers is usually told as a story of constant technical progress: in the 1990s, browser back and forward buttons took up half of the typical screen space; they choked on even basic, cranky GIF animations and plastered the screen with popup windows. Thanks to today's browsers we no longer need to download software, instead we can use smooth web applications, stream video, and communicate at lightning speed.[4]

However, the reality is more complex: from a software preservation perspective, not all changes in browser software have made each version "better" than the

original idea of interlinked documents. Browsers stopped supporting the plugins required for Java applets to run due to concerns about frequent crashes, slow performance, security problems, legal issues, and the availability of better-performing alternatives like Flash and JavaScript.

[4]The blog posts about the releases of new versions for Google Chrome or Microsoft Edge provide plenty of examples; see https://blog.google/products/chrome/ and https://blogs.windows.com/msedgedev/

previous one—just different. Central elements of a legacy site might be dependent on the capability that legacy browsers used to offer and which were later removed for a variety of reasons. Even if this was done with the best of intentions, archived websites will not be able to perform actions such as spawning a new window with a QuickTime movie playing and therefore might not make much sense at all anymore.

A classic example is the deprecation of the Flash plugin. Even in 2015, Adobe boasted that "More than 500 million devices are addressable today with Flash technology".[5] In 2020, Flash is just a faint memory. Websites based on the plugin display error messages or warnings (see Fig. 2).

Even features that are not usually considered in relation to the "rendering" of content can affect how it is perceived. Many web authors used to put meaningful messages in their pages' source (see Fig. 3), and the "View Source" function introduced with Tim Berners-Lee's first browser used to be heralded as one of the key factors for spreading knowledge about how to create webpages among users.[6] With powerful "developer tools" being offered in today's browsers, their ability just to show the basic HTML source code of a webpage is not getting as much attention from vendors as it used to. Functions for viewing source code are removed from menus or may even display garbled characters.

Ideally, archived websites should be accessed via a software environment that is contemporaneous with their creation. The project oldweb.today with its remote browser framework is offering exactly that.

3 Oldweb.today

On the website https://oldweb.today users can browse the past Web using browsers from the past, instantly and without any previous configuration necessary. The site is hosted by the digital art and culture non-profit Rhizome in New York and was started as an interim experimental software development project by Ilya Kreymer just before he joined Rhizome as an official employee. Rhizome's preservation director Dragan Espenschied designed the site's user interface and the online promotional campaign.

To get going, users have to pick a browser from a list (Fig. 4), provide a URL, and select a point in time (Fig. 5), before hitting the button "Surf the old web!" Users who do not have a clear interest here and just want to enter the past Web quickly can hit the button "I'm feeling random!" to be transported to a curated setting pulled from a list.

[5]See Adobe, Statistics, captured on 31 July 2015, https://arquivo.pt/wayback/20150408120146/http://www.adobe.com/products/flashruntimes/statistics.html

[6]One of many possible quotes: "The 'View Source' menu item migrated from Tim Berners-Lee's original browser, to Mosaic, and then on to Netscape navigator and even Microsoft's Internet Explorer. Though no one thinks of HTML as an open-source technology, its openness was absolutely key to the explosive spread of the web". (O'Reilly, 2005)

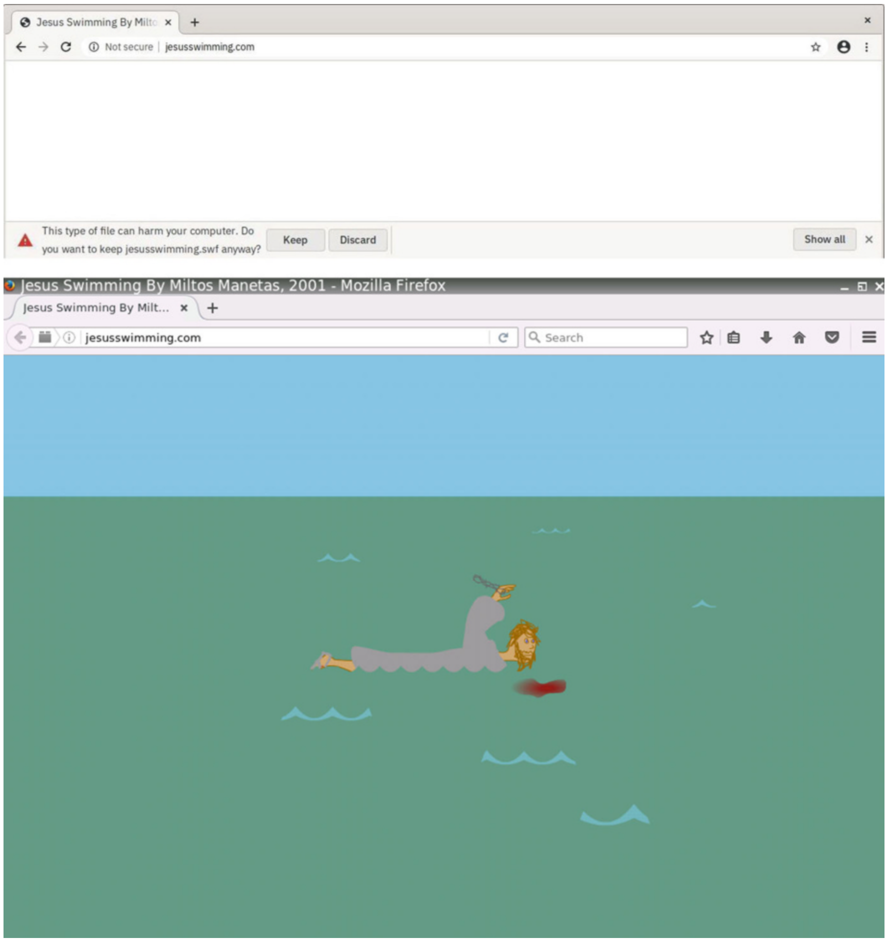
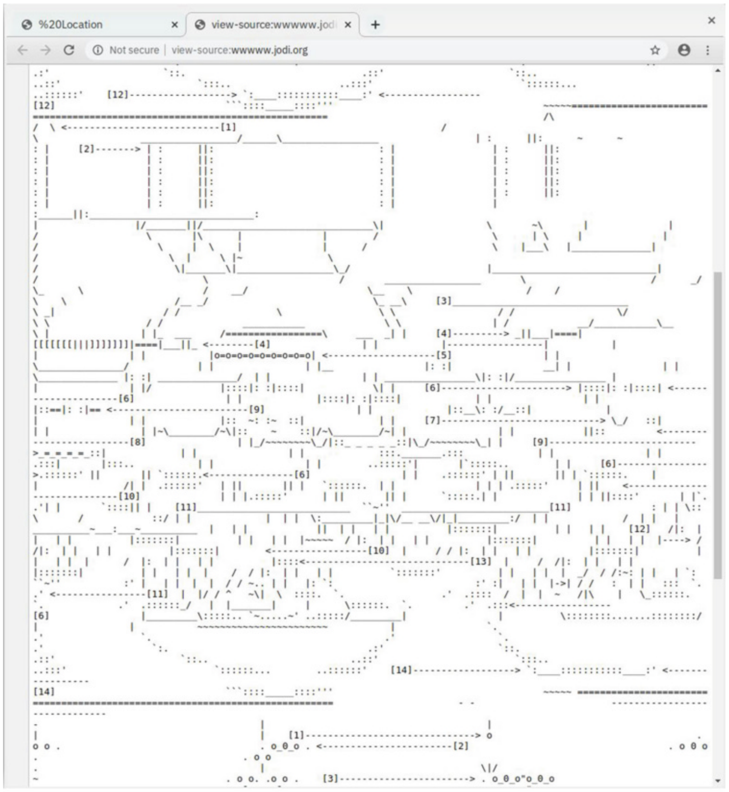

This type of file can harm your computer. Do you want to keep jesusswimming.swf anyway?
Jesus Swimming By Milto
Not secure | jesusswimming.com
Keep
Discard
Show all
Jesus Swimming By Miltos Manetas, 2001 - Mozilla Firefox
Jesus Swimming By Milt...
jesusswimming.com
Search

Fig. 2 Miltos Manetas, Jesusswimming.com, 2001, prompts Google Chrome version 80 (top) to display a warning in the user interface. In Mozilla Firefox version 49 with Adobe Flash plugin enabled (bottom) the work is displayed correctly. (Via oldweb.today.)

On the following screen (Fig. 6), oldweb.today establishes a video connection to a "remote browser", a carefully prepared, fully interactive software environment running the selected browser on a cloud computer. That remote browser is tied to a web archive aggregator that locates and pulls the requested materials from publicly available web archives, like the Portuguese web-archive (Arquivo.pt), the UK Web Archive, Rhizome's own web archive, and of course the Internet Archive. Information on all currently connected web archives is listed on the oldweb.today



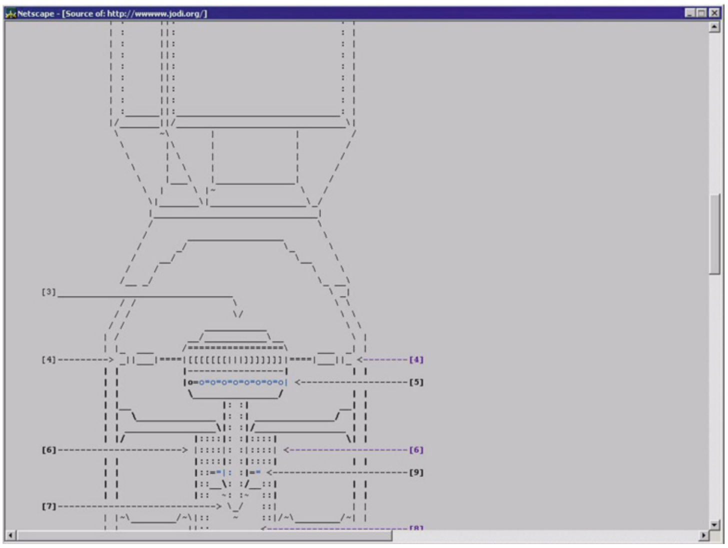


Fig. 3 Source view of JODI, %20Location, 1995, in Google Chrome version 80 (top) shows seemingly random characters. In Netscape Navigator Gold 3.04 for Windows (bottom), the source view of the same piece shows the schematics of an atomic bomb as ASCII graphics. (Via oldweb.today.)

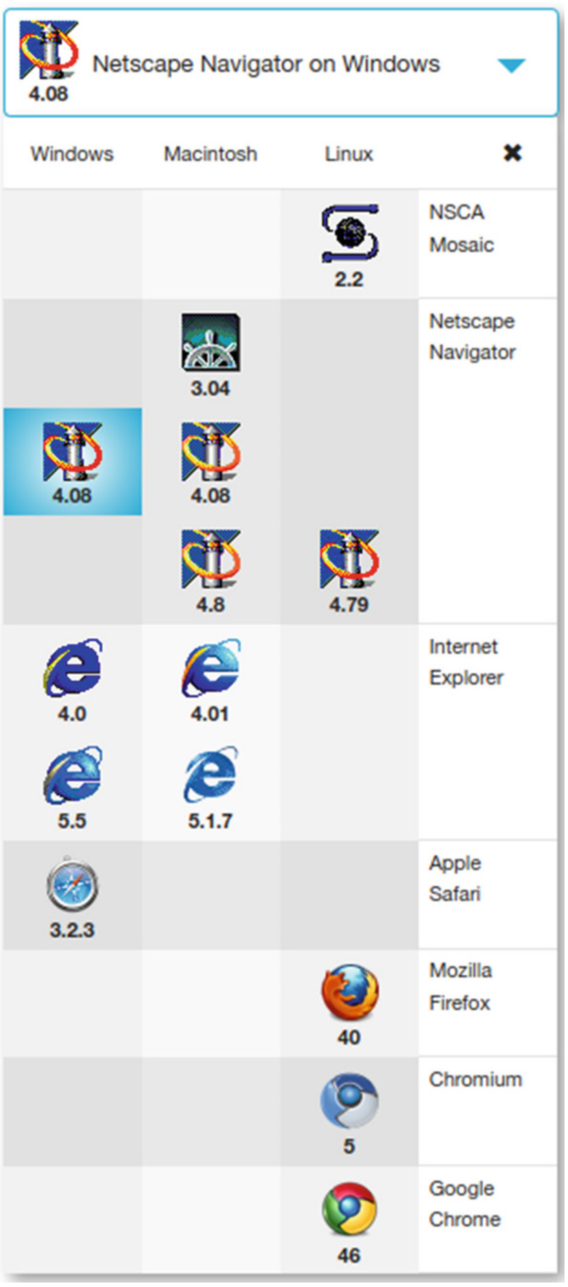


Fig. 4 The expanded browser selector available at oldweb.today

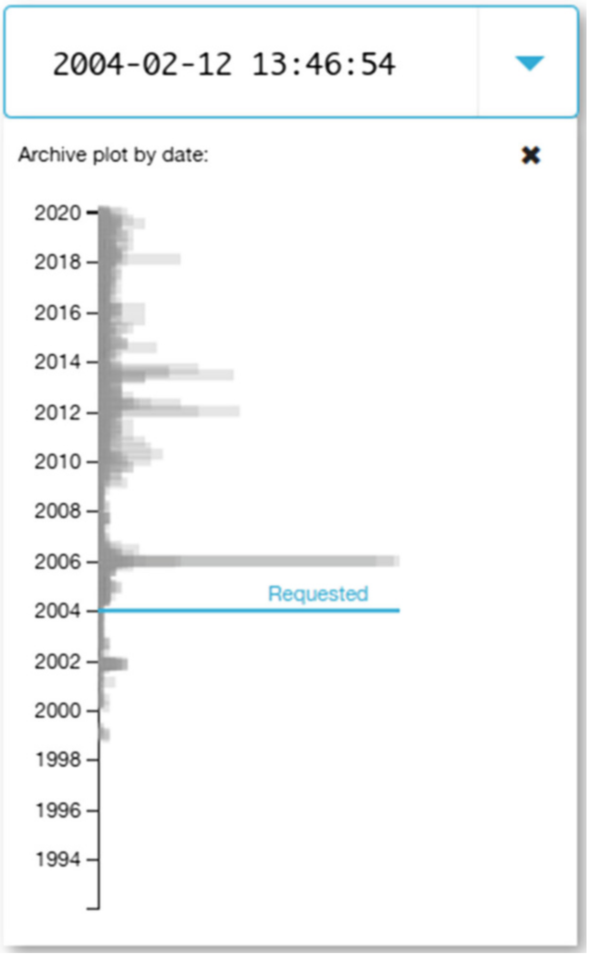
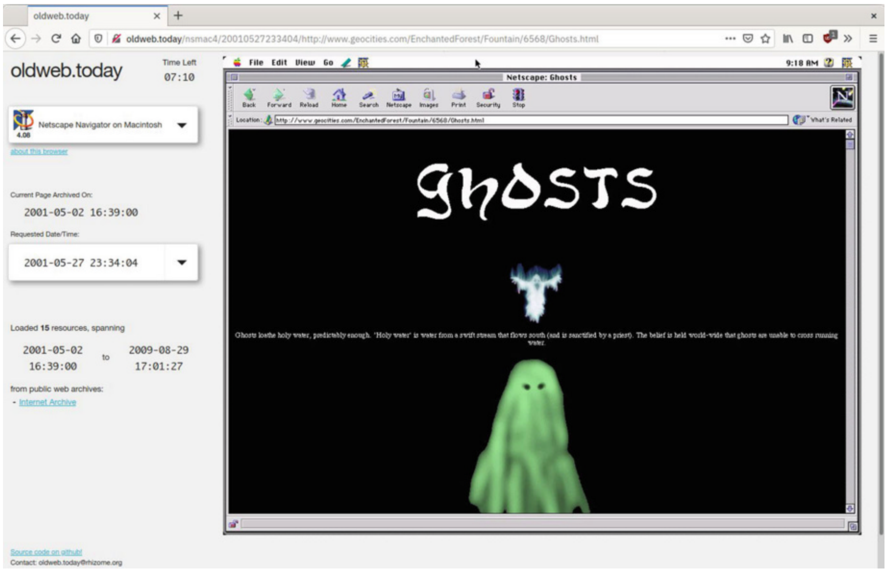


Fig. 5 The date picker on oldweb.today shows a graph of how many mementos of the requested URL are available across all connected public web archives

home page, and, with some more technical details, in a GitHub repository.[7] Provenance information for each accessed webpage is displayed on the side (see Fig. 7).

The oldweb.today site was launched with a set of 13 browsers, on 30 November 2015, running on Amazon Web Services, and "went viral" shortly afterwards, seeing over one million users in a few weeks of operation. For every user, a separate copy of a browser is launched. Since running a browser is much more computationally intensive than offering a typical web service, and cloud resources are billed by reserved computing capacity, oldweb.today features a time limit per session and a waiting queue to control the amount of concurrent usage and therefore cost.

[7] See https://github.com/webrecorder/public-web-archives



oldweb.today/nsmac4/20010527233404/http://www.geocities.com/EnchantedForest/Fountain/6568/Ghosts.html

Time Left 07:10

Netscape Navigator on Macintosh 4.08

Location: http://www.geocities.com/EnchantedForest/Fountain/6568/Ghosts.html

Current Page Archived On: 2001-05-02 16:39:00

Requested Date/Time: 2001-05-27 23:34:04

Loaded **15** resources, spanning 2001-05-02 16:39:00 to 2009-08-29 17:01:27

from public web archives:
- Internet Archive

Contact: oldweb.today@rhizome.org

Ghosts loathe holy water, predictably enough. 'Holy water' is water from a swift stream that flows south (and is sanctified by a priest). The belief is held world-wide that ghosts are unable to cross running water.

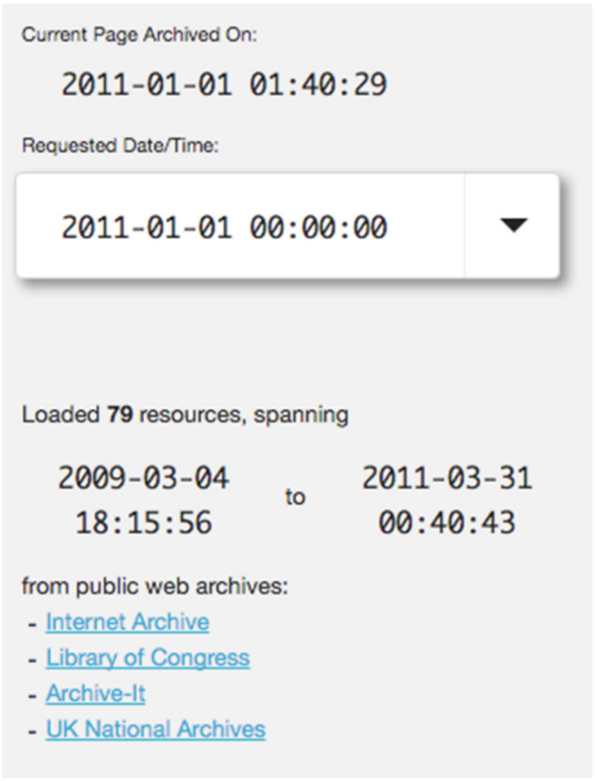
Fig. 6 Oldweb.today providing access to a legacy GeoCities website in Netscape 4.8 for Apple Macintosh

Rhizome promoted the free service with Jan Robert Leegte's net art piece *untitled[scrollbars]*,[8] a webpage created in 2000 that mainly consists of default scrollbars, consciously creating a drastically different look depending on the browser with which it is accessed (see Fig. 8).

In addition to oldweb.today's novelty and accuracy in reproducing what might too easily be dismissed as retro aesthetics, it offers significant digital preservation use cases. For example, the first US website published by Stanford University used the image format X-Bitmap, which was only supported in the Mosaic browser. Other browsers could render the HTML, but not this particular image (See Fig. 9).

Using a framework like oldweb.today effectively makes file format migration work redundant: legacy file formats like images and videos do not have to be transcoded to more current formats, and no intervention has to happen with the materials stored in web archives just so they can stay accessible. Instead, work can focus on a small number of browsers to remain available as running software.

[8]In 2010, the artist retitled the piece as *Scrollbar Composition*, http://www.scrollbarcomposition.com/. Discussed here is the version as found in Rhizome's collection.



Current Page Archived On:

2011-01-01 01:40:29

Requested Date/Time:

2011-01-01 00:00:00

Loaded **79** resources, spanning

2009-03-04 18:15:56 to 2011-03-31 00:40:43

from public web archives:

- Internet Archive
- Library of Congress
- Archive-It
- UK National Archives

Fig. 7 Detail of the oldweb.today interface showing some provenance information for different resources and web archives being used to assemble the currently visible page

4 Technical Excourse: Oldweb.today and Remote Browsers

It is possible to keep legacy software available for use in an archival setting (Suchodoletz et al. 2013). This can be confirmed by everyone who ever played a video game for a legacy system like the Nintendo Gameboy on their laptop using an emulator. Emulating early console games is comparatively simple, as the systems themselves had very few moving parts: the hardware of consoles did not change significantly during their time on the market, and the games were delivered on standard media like cartridges and CDs that only needed to be placed into the device to start a game. Console emulators mirror this architecture: a piece of software, the emulator, mimics the console device; an “image file” contains all the data that would be present on a game medium. Browsers are much more complex: they need an operating system that supports window management, Internet connectivity, font rendering, media playback, and much more. Running a legacy browser requires more than just storing an installer file for the software; instead, a complete software environment is needed. Such an environment usually requires expert knowledge to set up, using a general emulator or virtualisation tool.

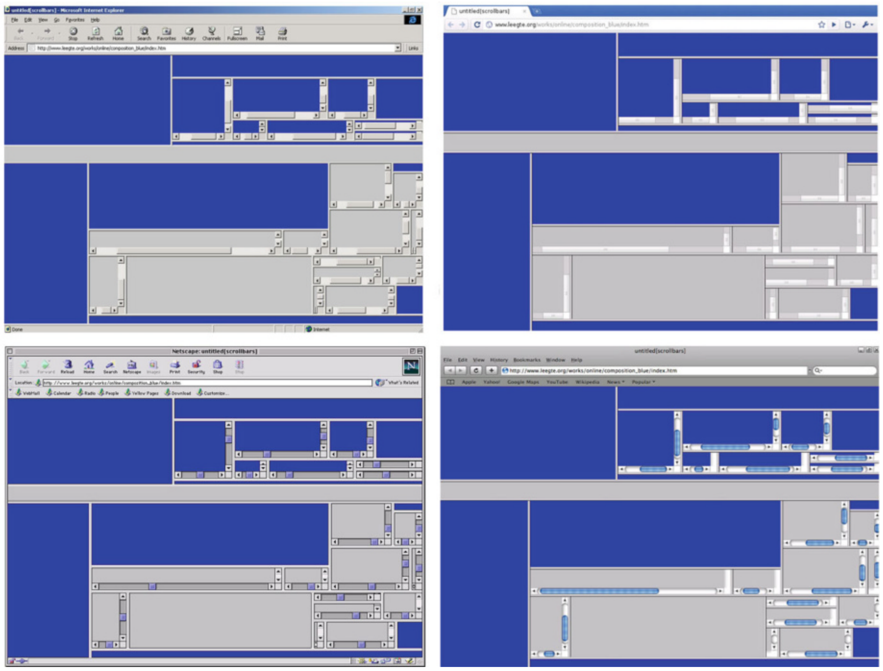
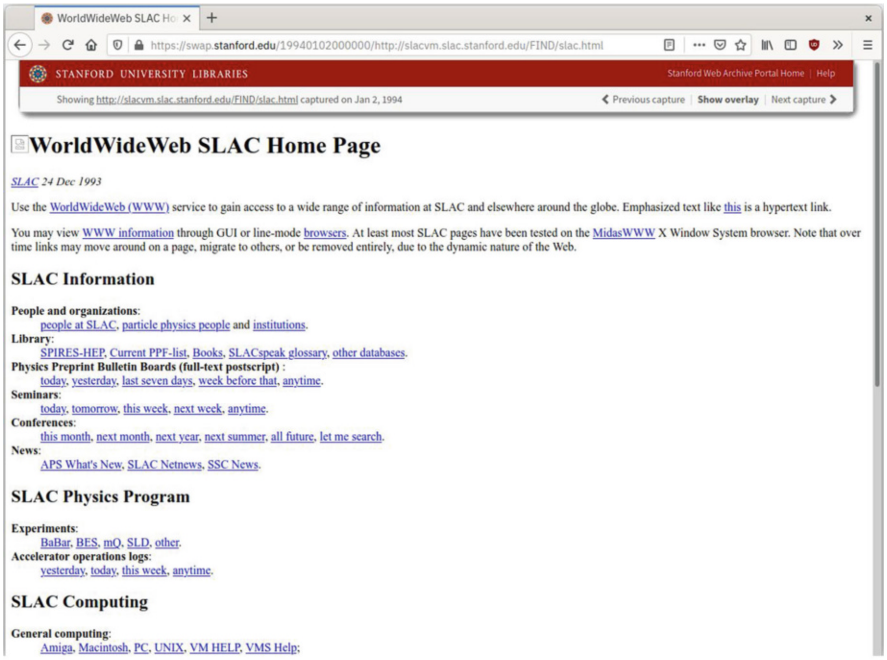


Fig. 8 Jan Robert Leegte, *untitled[scrollbars]*, 2000, accessed via Microsoft Internet Explorer version 4.0 for Windows (top left), Google Chrome version 5 for Linux (top right), Netscape Navigator 4.8 for MacOS 7.5 (bottom left), and Apple Safari version 3.2.3 for Windows (bottom right)

Oldweb.today packages these software environments in such a way that combining a web archive with a suitable browser is as easy as plugging a virtual game cartridge into a Gameboy emulator. This is in general possible because of two foundational features of the Web that have not changed very much: the HTTP protocol and connecting to the Web via a “proxy”. Regarding HTTP, even major updates such as encrypted HTTPS or speed-optimised HTTP/2 are just new wrappers around the same data being transmitted. Proxy server settings supported since the first browsers were released are still in use today. Within institutional settings, in particular, it remains common that the browser connects to the outside Internet via an intermediary computer. Hence, a web archival system that allows connection via proxy and can talk in HTTP will be able to serve almost any browser, past, present, and future.

Oldweb.today browsers themselves are running inside Linux containers. Containers are isolated configurations that allow very specific versions of software to be executed on a Linux operating system without clashing with other software that


WorldWideWeb SLAC Ho
https://swap.stanford.edu/19940102000000/http://slacvm.slac.stanford.edu/FIND/slac.html
STANFORD UNIVERSITY LIBRARIES
Stanford Web Archive Portal Home | Help
Showing http://slacvm.slac.stanford.edu/FIND/slac.html captured on Jan 2, 1994
Previous capture | Show overlay | Next capture

WorldWideWeb SLAC Home Page

SLAC 24 Dec 1993

Use the WorldWideWeb (WWW) service to gain access to a wide range of information at SLAC and elsewhere around the globe. Emphasized text like this is a hypertext link.

You may view WWW information through GUI or line-mode browsers. At least most SLAC pages have been tested on the MidasWWW X Window System browser. Note that over time links may move around on a page, migrate to others, or be removed entirely, due to the dynamic nature of the Web.

SLAC Information

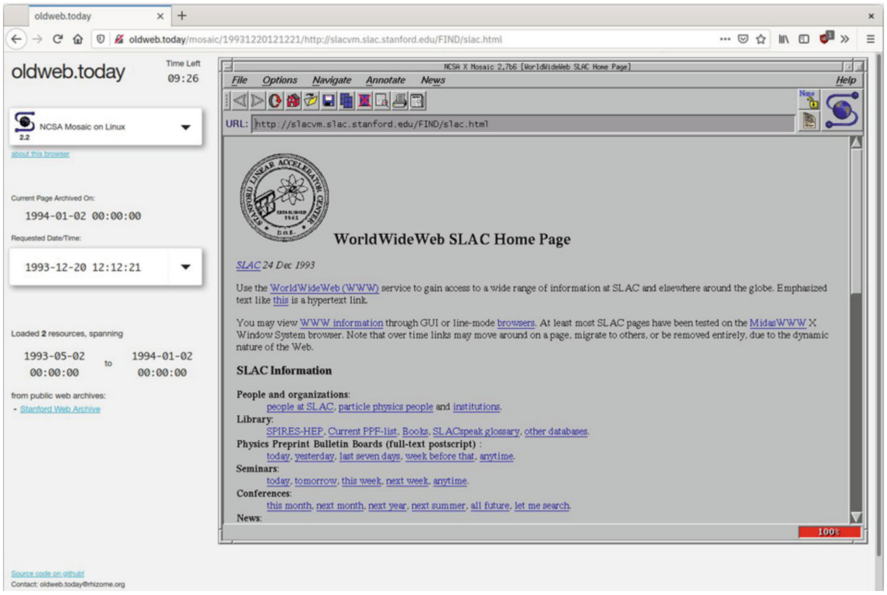
People and organizations:
people at SLAC, particle physics people and institutions.
Library:
SPIRES-HEP, Current PPF-list, Books, SLACspeak glossary, other databases.
Physics Preprint Bulletin Boards (full-text postscript) :
today, yesterday, last seven days, week before that, anytime.
Seminars:
today, tomorrow, this week, next week, anytime.
Conferences:
this month, next month, next year, next summer, all future, let me search.
News:
APS What's New, SLAC Netnews, SSC News.

SLAC Physics Program

Experiments:
BaBar, BES, mQ, SLD, other.
Accelerator operations logs:
yesterday, today, this week, anytime.

SLAC Computing

General computing:
Amiga, Macintosh, PC, UNIX, VM HELP, VMS Help;


oldweb.today
oldweb.today/mosaic/19931220121221/http://slacvm.slac.stanford.edu/FIND/slac.html
oldweb.today
Time Left
09:26
NCSA Mosaic on Linux
2.2
about this browser
Current Page Archived On:
1994-01-02 00:00:00
Requested Date/Time:
1993-12-20 12:12:21
Loaded 2 resources, spanning
1993-05-02 00:00:00
to
1994-01-02 00:00:00
from public web archives:
- Stanford Web Archive
Source code on github!
Contact: oldweb.today@rhizome.org
NCSA X Mosaic 2.7b6 [WorldWideWeb SLAC Home Page]
File
Options
Navigate
Annotate
News
Help
URL: http://slacvm.slac.stanford.edu/FIND/slac.html
100%

WorldWideWeb SLAC Home Page

SLAC 24 Dec 1993

Use the WorldWideWeb (WWW) service to gain access to a wide range of information at SLAC and elsewhere around the globe. Emphasized text like this is a hypertext link.

You may view WWW information through GUI or line-mode browsers. At least most SLAC pages have been tested on the MidasWWW X Window System browser. Note that over time links may move around on a page, migrate to others, or be removed entirely, due to the dynamic nature of the Web.

SLAC Information

People and organizations:
people at SLAC, particle physics people and institutions.
Library:
SPIRES-HEP, Current PPF-list, Books, SLACspeak glossary, other databases.
Physics Preprint Bulletin Boards (full-text postscript) :
today, yesterday, last seven days, week before that, anytime.
Seminars:
today, tomorrow, this week, next week, anytime.
Conferences:
this month, next month, next year, next summer, all future, let me search.
News:

Fig. 9 The 1994 SLAC home page accessed via the Stanford Web Archive's Wayback Machine, with the logo image missing (top). The same resource accessed via oldweb.today using an appropriate contemporaneous browser, Mosaic 2.2, correctly rendering the logo image (bottom)

might need a very different configuration to perform.[9] There are many frameworks freely available to handle containers. Oldweb.today uses Docker because it has been designed to efficiently package, distribute, and run preconfigured software environments and provides sophisticated networking features to connect these environments on demand—and because Docker is extremely popular among web developers and supported by major IT companies such as Google, Amazon, and IBM.

Several historic browsers such as Mosaic and Netscape are able to run directly in Linux containers because Linux versions of them were released in the 1990s. Supporting the much more widely used Windows and MacOS browsers required an extra step to be containerised. Old Macintosh browsers were installed in Linux versions of Basilisk II and SheepShaver,[10] the two free emulators most popular among classic Macintosh enthusiasts. Both of them can run MacOS version 7.5.3, which had been distributed by Apple free of charge, and subsequently provide versions of Netscape and Internet Explorer. A control panel to change the appearance of the operating system's default widgets was used to simulate the look of the later MacOS 8 where appropriate.

For Windows browsers, oldweb.today made use of WINE, an open-source software layer that imitates functions of the Windows operating system on Linux. WINE[11] is very popular among Linux users because it allows them to play Windows games or to use commercially released programs like Photoshop without having to purchase a Windows licence—but it can also run browsers. In the case of oldweb.today, the browsers deployed are several versions of Netscape and Internet Explorer and the Windows version that Apple released of their browser Safari, featuring the famous "brushed metal" design.

The browsers were chosen based on their public availability in archives (such as from the Evolt Browser Archive),[12] their historical significance, and their ability to replay legacy formats in a container setting. A few configurations took quite some effort to figure out, but since they are now packaged and released on the public Docker registry in one container per browser, this process will not have to be repeated. The whole stack is based on either open-source or free-of-charge software. If it can run on Linux, it can be squeezed into a container.

[9]Containers are based on core features of the Linux kernel and are used to set up server components on cloud services, distribute Android applications, run development environments, and much more. Organisations like the Open Container Initiative, https://www.opencontainers.org/, aim to create high-level specifications to increase interoperability between different container frameworks.

[10]Basilisk and SheepShaver are two popular open-source emulators of legacy Apple Macintosh platforms originally created by Christian Bauer. The source code is available on GitHub at https://github.com/cebix/macemu

[11]The WINE project provides a compatibility layer enabling Windows software to run on POSIX systems. See https://www.winehq.org/

[12]See https://browsers.evolt.org/

The web archive aggregator to which these remote browsers connect is based on the project "Memento Reconstruct"[13] which Ilya Kreymer previously implemented with the team at Los Alamos National Laboratory (LANL) (Sompel et al. 2009).

5 Using Remote Browsers to Capture Websites

Oldweb.today specifically focuses on historical browsers, bringing a new quality of access to historical materials in public web archives. Web archives of sites created today will soon become historical as well and face similar challenges of being optimised for outdated browsers. How can what we have learned from oldweb.today be fully integrated into common web archiving practice?

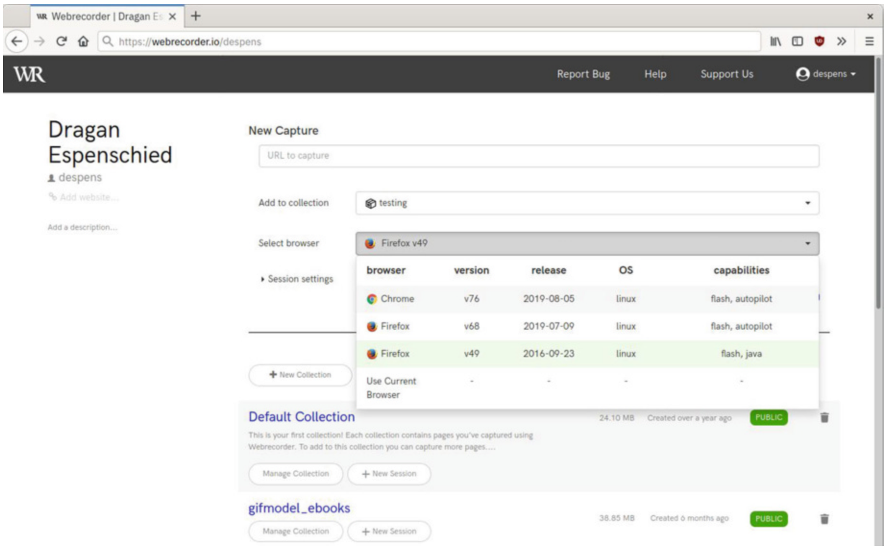
Oldweb.today is stewarded by the same team at Rhizome as the integrated web archiving platform Webrecorder.io,[14] which allows users to create a free account and capture web resources interactively by just browsing sites. Oldweb.today and Webrecorder.io are built using the same set of open-source software components; hence, it makes sense to offer users of Webrecorder.io remote browsers for capturing and accessing their own collections. Initially, Webrecorder.io only supported capture and access via whatever browser the user happens to visit the web service with. However, we noticed quickly that, for example, certain websites captured today using Chrome would already not be accessible in another browser such as Firefox. Websites operated by Google, in particular, might use experimental Chrome features like certain image formats, compression algorithms, or JavaScript extensions. On access, a browser other than Chrome would request other data, which would not be part of the collection. And, of course, there are also still plenty of websites on the live Web which are at risk of being abandoned and in need of saving precisely because they use plugins declared obsolete, mainly Flash and Java applets.

Under this premise, in October 2016, the remote browser framework powering oldweb.today was made into a separate component that could be integrated into the Webrecorder.io stack. Contemporary versions of Chrome and Firefox were preconfigured with Flash—and in the case of Firefox, with a Java 6 VM—and offered to users when starting a capture session. That session would be marked as being created with that browser, and, on access, the same configuration would be launched again (Fig. 10).

For example, this integration allows a user running Safari on their native machine to launch a version of Chrome to interactively capture in Webrecorder.io a particular site that contains Flash. Later, when another user running Firefox on their machine

[13] See https://github.com/ikreymer/memento-reconstruct

[14] Editor's note: At the time of publication, the web service Webrecorder.io had been renamed to *Conifer* and moved to https://conifer.rhizome.org. A new entity stewarding the software components was created as Webrecorder at https://webrecorder.net. The whole process is explained in a blog post at https://rhizome.org/editorial/2020/jun/11/introducing-conifer/



Dragan Espenschied
despens
Add website...
Add a description...

New Capture

URL to capture

Add to collection: testing

Select browser: Firefox v49

Session settings

browser	version	release	OS	capabilities
Chrome	v76	2019-08-05	linux	flash, autopilot
Firefox	v68	2019-07-09	linux	flash, autopilot
Firefox	v49	2016-09-23	linux	flash, java
Use Current Browser	-	-	-	-

Default Collection — 24.10 MB — Created over a year ago — PUBLIC

This is your first collection! Each collection contains pages you've captured using Webrecorder. To add to this collection you can capture more pages....

gifmodel_ebooks — 38.85 MB — Created 6 months ago — PUBLIC

Fig. 10 Before starting a capture session in Webrecorder.io, users can pick from a list of browsers with their special capabilities listed. To not overwhelm users, a small selection is presented for capture: current versions of Chrome and Firefox as well as browsers prepared with plugins. For access, any browser that was used for capture will stay available

accesses the collection, the same version of Chrome will launch remotely and run Flash. This works regardless of what the "local" browser might be (Fig. 11).

6 Looking Back and Looking Ahead

Using a browser released in 1996 cannot bring back resources that were not archived in 1996 but can make accessible resources available from that time in a much more authentic way (Espenschied et al. 2013). It has become more viable to use real browsers to capture content from the live Web for archiving using for instance Browsertrix.[15] As a result, it is hoped that web archives created today will appear more complete when accessed a decade from now, than web archives that have been built using custom crawlers that could never imitate all the features of a full-on browser.

The collecting, storing, and running of browsers will become easier in the future: as browser vendors use tightly structured and standardised release and installation processes, and Linux has become a common target platform, new browser versions could be "containerised" on the day they are released. It might be possible to execute

[15] https://github.com/webrecorder/browsertrix

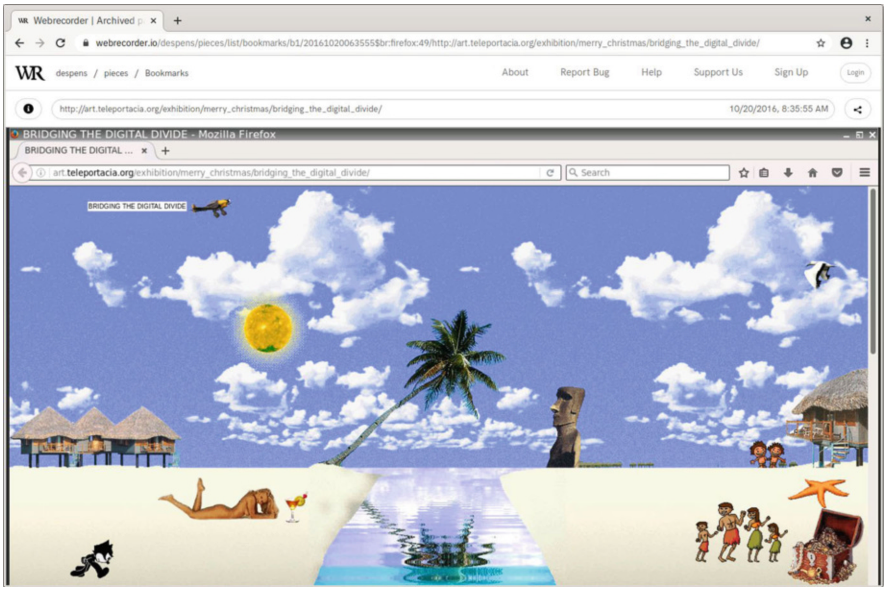

Webrecorder | Archived p
webrecorder.io/despens/pieces/list/bookmarks/b1/20161020063555$br:firefox:49/http://art.teleportacia.org/exhibition/merry_christmas/bridging_the_digital_divide/
despens / pieces / Bookmarks
About
Report Bug
Help
Support Us
Sign Up
Login
http://art.teleportacia.org/exhibition/merry_christmas/bridging_the_digital_divide/
10/20/2016, 8:35:55 AM
BRIDGING THE DIGITAL DIVIDE - Mozilla Firefox
BRIDGING THE DIGITAL ...
art.teleportacia.org/exhibition/merry_christmas/bridging_the_digital_divide/
Search
BRIDGING THE DIGITAL DIVIDE

Fig. 11 A copy Dragan Espenschied, Bridging The Digital Divide, 2003, captured and accessed with remote browser Mozilla Firefox version 49 on Linux including the Java plugin, integrated into Webrecorder.io

browsers that are old enough not to require much computing power via emulators delivered in JavaScript, running on a user's local computer rather than in the cloud—a browser itself being the host for one its ancestors, with the possibility of reducing the costs of bandwidth usage and computing. It is likely, however, that a powerful cloud computer would be required to run more recent browsers, making the remote browser concept pretty universal.

Over the past 5 years, remote browsers have demonstrated the possibility and benefits of accessing web archives in an alternative context: through stabilised, emulated browser environments contemporaneous with the archived web content, instead of the traditional "Wayback Machine" style replay. As the Web ages, and once common software like Flash becomes obsolete, running older browser software in emulators may be the only way to correctly view and interact with the old Web.

Preserving browser software is thus as important as preserving the Web itself to ensure future access. The still largely separate discipline of software preservation has to become an integral part of web archiving; not only because historic browsers need to remain available for sentimental purposes but also because the Web itself is transmitting software, not static documents.

References

Espenschied D, Rechert K, von Suchodoletz D, Valizada I, Russler N (2013) Large-scale curation and presentation of CD-ROM Art

O'Reilly T (2005) The open source paradigm shift. In: Wynants M, Cornelis J (eds) How open is the future? Vubpress, Bruxelles, p 102

Sompel H, Nelson M, Sanderson R, Balakireva L, Ainsworth S, Shankar H (2009) Memento: time travel for the web

Suchodoletz Dv, Rechert K, Valizada I, Strauch A (2013) Emulation as an alternative preservation strategy – use-cases, tools and lessons learned. In: Horbach, M. (Hrsg.), INFORMATIK 2013 – Informatik angepasst an Mensch, Organisation und Umwelt. Gesellschaft für Informatik e.V., Bonn (S. 592–606)