Rhizome proposes a two-year collaboration, beginning May 2016, with University of Freiburg (Germany) and Yale University to create new software tools to determine and manage software environments for the purposes of lowering the barrier for archives to adopt emulation as a digital preservation strategy.

Born-digital artifacts are most legibly preserved by being made *usable* again within the complex, interdependent software environments in which they were created. To this end, emulation is quickly becoming an attractive preservation strategy for memory institutions stewarding collections of legacy artifacts.

What, then, is holding emulation back? A practical emulation strategy requires knowledge about what pieces of legacy software (e.g., operating system and web browser) must be combined for a born-digital artifact to "perform," to behave as expected when code is run and respond as expected to external input. More, this knowledge and the actual executable software and complete environments—a combination of operating system, components, and applications—need to be managed in an efficient manner to meet a diversity of artifacts.

There is a gap between emulation frameworks and existing archives of operating systems and software. Presently, archives store and describe software as static objects and brush over performance aspects, or archives over-conceptualized software in complex ontologies or description systems that aim to understand software at a level too detailed and laborious to keep up with development pace or react to diversity and scale. Currently, there is no infrastructure responding to this gap; needed software is distributed across the grey market; and there is no system to register in-depth knowledge of standard environments that would represent "typical" computing setups within a certain era and domain, capable of performing many born-digital artifacts. As such, emulation is just not yet at a risk-level where many institutions can experiment with it.

This proposed project responds to the disparity between the proven viability of emulation as a digital preservation strategy and the practical needs of collection managers. Its primary goal: creating a tool to characterize and re-enact the contents of digital collections, usable as a local copy and a hosted service, acting as a facade to a curated software library.

The collaborators will define functional requirements for an interface between software archives and emulation frameworks, and define and design a metadata framework for this new link. The knowledge collection process will be practice-based, as it will allow users to evaluate the actual performance of software running inside emulated environments, and then generalize this evaluation to serve others. For example, if a combination of certain software performs a characterized artifact well (e.g. "web-site with Real-Video and Quicktime embeds from 1999"), it can be assumed that this software will be a good match for similarly characterized artifacts, and can then be deployed with confidence by others. Rhizome's ArtBase, a diverse collection of 2,000+ born-digital artifacts, will serve as a test-bed.

To describe the performance and capabilities of software, a tool will be developed that allows one to evaluate software in action, while it is running inside an emulation framework from a curated software archive. The metadata collected with this new workflow will be based on tested and observed behavior, cultivating valid statements about performance and the capabilities of combinations of software, knowledge that can be grown and refined over time via continued usage.

To foster the adoption of developed protocols and processes, any source code will be published and made available on open-source licenses. Information about the project will be communicated to both library and institutional professionals, with the aim of increasing adoption of emulation as a digital preservation strategy, and more broadly to the public via live events presented by Rhizome.

Rhizome's unique collection, mostly net art from the 1990s to the present, allows this project to evade licensing issues, since the involved software—web browsers, plugins, "multimedia" extensions and players—have always been distributed free of charge. And yet, the complexity is sufficient to gain insight into the structural relationship of software at large, including pieces with restrictive licenses.

I. Statement of Need

Born-digital artifacts are most legibly preserved by being made *usable* again within the complex, interdependent software environments in which they were created. To this end, emulation is quickly becoming an attractive preservation strategy for memory institutions stewarding collections of legacy artifacts. Emulation is also becoming more viable, with the proliferation of frameworks like Olive, Emularity, oldweb.today, and EaaS/bwFLA that enable these artifacts to be experienced within software contexts contemporaneous to a given artifact's production.

Practical integration of emulation into collections has already been successfully demonstrated for CD-ROMs at the transmediale[1], the Rose Goldsen Archive at Cornell[2], and the German National Library[3]. While CD-ROMs are relatively simple and self-contained—with regard to specific software dependencies—these case studies made clear that a performance-based, structured, distributed, and practical approach to managing dependencies of legacy systems would be a necessary precursor to the wide adoption of emulation as a preservation technique.

A barrier to adopting emulation as a preservation and access strategy is the problem of "software curation." Memory institutions wishing to experiment must ascertain for each artifact what kind of (emulated) computing environment is needed to make the artifact "perform" (that is, behave as expected when code is run and respond as expected to external input); what (legacy) software is required to build that environment; and how the software dependencies and resulting environments can be managed efficiently. Software curation is the practice of answering these questions by connecting the right pieces for an artifact to perform.

While it can be assumed that the performance fidelity of emulators and both the availability and the usability of emulation frameworks will steadily increase as models for their development have already been established, there remains a discrepancy between the emulation technology and the practical needs of collection managers[4]. To advance emulation as a premier preservation and access strategy, supporting infrastructure is needed that can build, manage and preserve software environments, and enable knowledge exchange and reproducibility—that is, what is needed is a tool for software curation.

The major building blocks necessary for this infrastructure—format characterization tools and registries, metadata standards, various software collecting entities and archives—already exist, yet remain isolated and lack a coherent strategy with regard to their integrated use for emulation. This project will meet the aforementioned need by creating a tool to close that gap on a practical, tangible level, advancing and integrating already available state-of-the-art emulation, software-archive, and metadata management practices.

This project relates to and develops previous work in academia. A key project is KEEP[5], an earlier undertaking funded by the European Union. Given a digital object, e.g. a Word Perfect document, the KEEP

---

[1] Dragan Espenschied, Klaus Rechert, Isgandar Valizada, Dirk Von Suchodoletz and Nick Russler, *Large-Scale Curation and Presentation of CD-ROM Art*, iPRES 2013. p.1-8

[2] https://ecommons.cornell.edu/handle/1813/41368

[3] K. Rechert, T. Liebetraut, O. Stobbe, I. Valizada, T. Steinke: Characterization of CDROMs for Emulation-based Access, 12th International Conference on Digital Preservation IPRES 2015, 2015

[4] See: David S.H. Rosenthal: "*Infrastructure for Emulation*", Blog Post, September 8 2015

[5] Keeping Emulation Environments Portable (KEEP), http://www.keep-project.eu/

Emulation Framework provides a file characterization workflow to determine a suitable "view path"[6] for rendering the object. The selection of the view path is based on an instance of the TOTEM metadata model[7], which was developed alongside KEEP. A view path, however, is a "mono-dimensional" representation of an emulated rendering environment, that is only able to map one set of dependencies to one single "file format" one-on-one; this limits the re-usability of any viewpath only to absolutely identical, isolated files.

Furthermore, TOTEM, as well as the recently developed PREMIS[8], provide metadata language / encoding to describe a computational context (e.g. emulator setup and software dependencies) in an abstract way. The production of such environments and their (usable) encoding into metadata is left unresolved, as is (auxiliary) software that can recognize a suitable emulated computing environment that an artifact requires and then connects these two to re-enact the artifact and facilitate user interaction. KEEP, TOTEM and PREMIS haven't had much impact in lowering the barrier to adoption for emulation since the creation of view paths is very laborious and requires significant expert knowledge.

This proposed project seeks to extend these and other characterization and software dependency description concepts by applying them to multi-file and multi-format objects and by providing practical, easy, and functional access to preserved software components. As knowledge about the *performance* of software is about behavior and reaction, practical emulation requires dynamic technical metadata about how software functions—tested and refined by usage, review, and maintenance of that software—rather than static descriptive metadata (akin to bibliographic information).

But even when suitable computing environments and software dependencies have been identified as needed to perform a legacy artifact, getting the software components and the artifacts to perform together isn't supported by existing infrastructure. Although software archiving is a recurring research topic within digital preservation communities, little effort has been put into making software collections accessible and usable, leaving only the grey or black software market as a last resort, in particular for end-of-life legacy software. Memory institutions requiring emulation to access and render their digital artifacts therefore face further difficulties: available software packages are scattered over various hobbyist or community-driven collections and, depending on the original source, are provided in various forms and variations (zipped archives, images of installation media, etc.). This results in an unnecessary and laborious process to build a usable emulated rendering environment for each artifact.

Time and resources spent researching and building environments, moreover, is usually lost quickly as these curatorial processes are almost impossible to reproduce consistently due to a lack of structured formats to describe them. From the aforementioned projects involving large CD-ROM-based collections, it became clear that a systematic approach to emulation is key to preventing redundant work, collecting fading knowledge, and enabling emulation to scale to larger collection sizes.

Software dependencies, or even more complex relationships, are better "captured" than described, by a tool that watches how an emulated environment is created by a curator and is able to abstract activity into re-usable, machine-actionable metadata. Preservation planning, with regard to ensuring long-term

---

[6] J. van der Hoeven and D. von Suchodoletz. "Emulation: From digital artefact to remotely rendered environments." *International Journal of Digital Curation*, 4(3), 2009

[7] J. Delve and D. Anderson. "The Trustworthy Online Technical Environment Metadata Database – TOTEM." Number 4 in *Kölner Beiträge zu einer geisteswissenschaftlichen Fachinformatik*. Verlag Dr. Kovac, Hamburg, 2012.

[8] A. Dappert, S. Peyrard, C. C. Chou, and J. Delve. "Describing and preserving digital object environments." *New Review of Information Networking*, 18(2):106–173, 2013.

access, should be focused on a small set of emulated computing environments (a so-called "standard environment" mimics a typical computer setup—hardware and software/applications—of a given artifact's era), since these environments could be re-used for a large number (thousands) of artifacts. This project aims to improve the re-usability and reproducibility of such environments and to reduce management and long-term preservation costs.

The institutions collaborating on this project—Rhizome, University of Freiburg[9], and Yale University—have gained deep insight into emulation at scale through past individual research and partnerships. University of Freiburg, creators of the bwFLA framework[10], have worked with the German National Library, and, with Dragan Espenschied, on preserving the Vilèm Flusser Archive and the transmediale archive. When Espenschied joined Rhizome in 2014, the cooperation continued, shifting from large collections to a large user-base, with legacy born-digital artifacts presented on the public web using cloud computing. Rhizome has also created the service oldweb.today, which connects public web archives with specialized emulation environments. Yale University was an early adopter of bwFLA and has, in close cooperation with University of Freiburg, undertaken significant steps to integrate emulation into their immense digital collection, with diverse holdings produced from the 1980s to today. All three institutions are partners on a bilateral NEH/DFG research project to create a citation system for digital artifacts in their executed state.[11]

As a related project, the *Software Preservation Network*[12] is a promising initiative to create a network of institutions that would collect and share software. Their focus is very broad, targeting, among other things, community-building and working on legal/licensing issues that are currently perceived by most institutions as the biggest hold-ups to adopting emulation. We are in conversation with the SPN and see this project as an infrastructure component the initiative might adopt. Their plan to draft architectural specifications that consider integration with existing national preservation infrastructure is a possible platform for communicating the approaches this project seeks to promote.

II. Impact

This project aims to lower the barrier to adopting emulation as a preservation strategy by providing a structured approach to managing software and emulated environments that enables the execution, or performance, of born-digital artifacts.

This project will produce a tool to characterize and re-enact the contents of digital collections, usable as a local copy and a hosted service, acting as a facade to a curated software library. All software and data sets created during the project will be released under permissive free and open source licenses, including a road map for institutions that want to adopt emulation or offer emulation services. The project seeks to contribute to the establishment of an approach to documenting software, disk images, and emulator configurations with the aim to move towards a standard workflow and technical process for use by

---

[9] Please note that contributions by University of Freiburg will be independently funded. While included in the work flow and deliverables below, this aspect of the project will not be funded by the IMLS grant.

[10] See: http://eaas.uni-freiburg.de/

[11] "Tools and Concepts for Safeguarding and Researching Born-Digital Culture–(Re-)Defining Object Boundaries and Citation of Complex Digital Objects", granted in May 2015 by the DFG/NEH Bilateral Digital Humanities Programme

[12] See: SPN proposal to the IMLS, grant awarded October 2015

the wider community. All participants will work in dialogue with the digital preservation community, connecting with bodies like the newly forming Software Preservation Network (SPN), as well as digital conservation practitioners working in the arts and libraries.

To keep the project focused and offer an easy path to adoption, we seek to link technical metadata to software objects as an independent layer, instead of as an integration into existing archives or the building of yet another registry. File-based, long-term digital preservation (bitstream preservation) is an established practice and works well, but is very different from the dynamic usage and maintenance of files that happen to be executable software. Separating static and dynamic metadata allows one to request a file or disk image from a classic digital repository, inject it into an emulation framework, execute it there, and collect data about its execution. Since this data is likely to be refined and extended every time the software is executed in a different environment, it should be managed separately.

Since both the construction and the preservation of emulation environments generate costs, it is most efficient to keep as few environments as possible that are as versatile as possible. This project aims to help institutions understand and evaluate existing collections of born-digital artifacts, providing information on the desirable components of execution environments for those artifacts.

Digital artifacts are inherently tied to computational performance and interaction inside complex systems. The complexity even for just displaying a seemingly simple single text file can be overwhelming to first fully understand and secondly describe. It can be said that previous attempts to gather information on software have been heavy on a "describing before using" paradigm and weren't able to generate machine-actionable metadata that would allow creating execution environments or matching them to artifacts effectively. Hence, there have been little practical results. To orient this project towards applicability, we will work to reach verifiable goals:

Based on a limited initial collection (Rhizome's ArtBase), a functional and usable software archive will be created. Given this initial software archive, we want to ensure that it is usable and the software functional at different sites/institutions (Rhizome and Yale University); both the technical metadata and the software pieces will be organized for meaningful, practical re-use.

Given an environment that has been constructed for an initial collection of artifacts (Rhizome's ArtBase) with metadata-enriched software and therefore has known capabilities, we aim to classify and use similar yet unknown artifacts (at Yale University) with the same environment. This requires for the environments to be reasonably portable and their metadata applicable to re-construct an environment.

As an optional goal we want to identify new partners that would allow functional access to their software collections.

Even if emulation and bitstream conservation of artifacts and required software works perfectly, there is still a constant loss of knowledge about **how** software has to be installed and used. Information that is good at the release time of software ("requires 14" color screen and VGA graphics adapter") becomes useless pretty quickly: none of these components are easily available today, or actually matters much for the execution of software. Further, the operation of a certain software might not be apparent to future users; implicit knowledge that is crucial to make a software perform correctly is hardly ever described or recorded.

This is why technical metadata about software must be collected practice-based, and be integrated into an emulation framework, immediately showing the failure or success of a software combination. Hence, every claim in the resulting dataset will be based on actually performing systems and readily available software.

A decentralized approach to collecting the technical metadata will allow domain experts and eyewitnesses to formalize their knowledge about how software is and was actually used. For instance, Yale University holds a large collection of *AutoCAD* files, a highly complex software package typically used by

experts. Knowledge located in this institution will be able to evaluate if *AutoCAD* performs as intended in an emulated environment, other institutions will then be able to rely on this information, and benefit if the information is refined later. A simpler, more distributed example is the common recommendation to use Adobe's *Acrobat Reader* to render PDF files, while real-world users try to avoid this software since it is resource-hungry, crash-prone, and connected to an annoying updating mechanism. Alternative PDF readers have gained large popularity and again in certain domain-specific cases and technical contexts might be the actual correct way of rendering, regardless of "feature completeness".

III. Project Design

The project is designed around three key assumptions:

1. Emulation as a tool to provide access to archived digital artifacts "works" with sufficient fidelity. We will target only the level of software and application above the operating system level.[13] Evaluations of detailed emulation performance—for instance, how legacy hardware like a CRT monitor influences the look of pixel graphics and how the emulator would reproduce that—is out of project scope, since this would attract feedback on the underlying emulator or framework, not on the relationships of software.
2. Storage (stream preservation) as a process "works". Storage and Usage & Maintenance of software can be meaningfully separated; bit-stream preservation and cataloging is about fixing a piece of software in dormant state, while data about its operation and capabilities is subject to dynamic change as it gets executed in varied contexts. In practice this means that knowledge about how software is to be used and operated can not be expected to be part of a static catalog.
3. The licensing issue is "unsolved" and this project doesn't aim to present a comprehensive solution for managing and running software with license restrictions. For pragmatic reasons, this project will work only with software that doesn't pose restrictions on its usage and distribution, as is the case with web browsers and related plugins that have been in wide general use. While this use-case offers a rather small and closed set of different software components and is mostly risk-free regarding software licensing ("free-as-in-free-beer" software), it still offers enough complexity to provide new and valuable insights into technical and organizational issues to be applicable to any kind of software later.

The project is about establishing a connection between:

- Existing emulation frameworks that have been proven to deliver results; since all participants have the most experience with bwFLA, this framework will be targeted in technical production, while the project's general outcome will be framework agnostic.
- The existing PRONOM[14] data format registry and its surrounding tools (DROID, Siegfried[15]); this well-established platform provides high quality identification of single files. PRONOM will be used to describe the capabilities of software and emulation environments.

---

[13] For a discussion of emulation layers, see: Dirk von Suchodoletz, Rechert Klaus and Bram van der Werf, "Long-term Preservation in the Digital Age – Emulation as a Generic Preservation Strategy", *PIK* 35, (4), 2013 De Gruyter. S.225-226, DOI: 10.1515/pik-2012-0051

[14] See: http://apps.nationalarchives.gov.uk/PRONOM/

[15] See: http://www.itforarchivists.com/siegfried

- Existing public collections of software, or collections accessible to the project partners; enthusiast web sites like evolt.org, amateur freeware libraries like oldversion.com, the Internet Archive as well as Rhizome's and Yale University's internal collections of software, while not structured for Usage & Maintenance provide enough material to serve as exemplary collections.

Missing pieces this project aims to develop are:

- A tool and workflow to identify a suitable emulation environment (emulator, operating system, libraries, and applications) for a given artifact, where the artifact can be composed of multiple, interrelated parts or even represent a collection. If a match is not available, the tool will present information about the desired components of new environments that need to be constructed, and possibly suggest an environment that is missing as few components as possible to base the new environment on.
- A tool and workflow to manually create and evaluate instances of emulation environments that allows the structured installation and test of software components in combination with artifacts. The results will be usable, running environments and metadata that describes their capabilities (so they can be used on artifacts with similar characteristics) and the activities leading to their construction (so they can be reproduced/re-constructed). In particular, the metadata will be technologically neutral, to be used with different emulation frameworks, but most importantly with future emulators.
- A distributed, dynamic metadata layer for recording information about capabilities, dependencies, and availability of pieces of software; this also requires a technical specification on how this layer can serve as a facade for existing software collections and collection management systems.

In contrast to earlier research[16], we do not aim to construct full software ontologies or to completely "understand" software: it is simply too complex of an undertaking that has yet to show tangible benefits. Recording the minimal metadata required for the project, we will focus on what we have identified as core elements of software: its performance, capabilities (the data types it can handle), and its relationships to other software.

Since it is assumed that knowledge about the operation of software is largely implicit and based on the software being performed rather than described, the metadata collection will be designed to be usage-driven as much as possible, to structure the preservation of *performance*, not descriptions of software as static objects.

This project's main use-case is providing authentic access to complex, archived web sites via emulation environments with capabilities similar to those typically used in the epoch the websites were created.

For instance, a 1998 website containing HTML (PRONOM fmt/96), interactive VRML 3D models (PRONOM fmt/94), and JPEG images for textures (PRONOM fmt/44) cannot be viewed with arbitrary environments or "any modern browser". While for example there are almost endless possibilities for software that can handle the display of an isolated JPEG file, to produce the artifacts' actual meaning, an environment is required that can work with all three listed data formats in combination. Possible matches could be a Windows XP environment using the Cortona plugin for Internet Explorer 6, or a Windows 98 one with Netscape 4.7 and Comso Player 2. Since the epoch of the artifact fits the second environment better, it would become the top recommendation.

---

[16] See: KEEP and TOTEM, as previously discussed in this document; and: Brian Matthews, Arif Shaon, Juan Bicarregui, Catherine Jones: "A Framework for Software Preservation**."** http://dx.doi.org/10.2218/ijdc.v5i1.145

The evaluation tool will allow archivists, librarians, and digital preservation practitioners to immediately verify and feedback on these suggested alternatives by injecting the artifact into the suggested emulation environments, using the Emulation as a Service framework. If no environment can be matched, the tool will allow for the creation of a new one, listing already configured environments that come closest—in this example, there might be one that is just missing a VRML plugin.

In a second example, a technically simple website, created in 2000, only transports its meaning as intended if scrollbars are displayed by the operating system as common user interface elements. This is not the case for today's mobile operating systems, with the design of desktop operating systems catching up: Mac OS X has not shown scrollbars by default since 2011, and many users don't even know what scrollbars were, what they looked like, and what purpose they served.[17] By matching the artifact's era, an emulation environment providing the desired performance (in this case: scrollbars) will be recommended without the need for further knowledge at time of access or guesswork at the time of the artifact's creation.

This example illuminates how born-digital art provides many edge-cases and challenges assumptions about digital preservation in ways that have been extremely productive for the whole preservation field.[18] Learnings from this approach provide groundwork for other, similar tasks like research data management. In order to verify results not only data is required but an authentic and complete software environment rebuilding the researchers workflows.

IV. Project Resources: Personnel, Time, Budget

Rhizome's lead digital conservator, Dragan Espenschied, will oversee the project (20% of his time), and additionally Rhizome will hire a software curator (80%). Yale University's Euan Cochrane will oversee Yale's part (5%) and have their software curation CLIR fellow (32%) work on the project. Dr. Klaus Rechert, lead of the bwFLA project at University of Freiburg, will oversee development and have a developer with a specialization on emulation work on implementation.

Please note that contributions by University of Freiburg will be independently funded. While included in the work flow and deliverables below, this aspect of the project will not be funded by the IMLS grant.

There are four interwoven work packages that have been designed to fit the institutions' core competencies, described in detail in the supporting document "**Detailed Work Plan**".

Rhizome provides resources for edge-case heavy, diverse artifacts from its ArtBase collection, leading artistic research competence, and experience having already adopted performance-based archiving at the core of its institutional practice. Yale University provides deep knowledge about state-of-the art archival and collection management systems, systems operations, and experience about institutional implementation of standards and best practices and how they hold up with very large collections. Both Rhizome and Yale are heavily invested in using emulation as a preservation strategy.

University of Freiburg is developing the versatile and powerful emulation framework bwFLA, which is already used internally and/or public-facing at both Rhizome and Yale; personnel at Freiburg will be responsible for all implementation work. bwFLA has been combining technical excellence with a focus on preservation planning with emulation from the start.

---

[17] This case was publicly executed online by Rhizome with an art piece by Jan Robert Leegte, see: Rhizome: "Cyberspace, the old-fashioned way," blog post, 3 Nov. 2015

[18] See: Trevor Owens: Digital Art Curation Grad Seminar: Your Input Welcome, Blog Post, August 29 2015

WP.1 Concepts and Infrastructure planning

1.1 Set up local infrastructure for a software archive after evaluating different options (Rhizome/ Yale)

1.2 Define functional requirements for an interface in between software archive and emulation framework (Rhizome/Freiburg/Yale)

1.3 Analyze and define requirements for additional metadata for software to enable structured use within the emulation framework (Freiburg/Rhizome)

WP.2 Prepare Use-Case, Test & Evaluation

2.1 Provide use case and systematically ingest software (Yale)

2.2 Prepare artifact collection (ArtBase) to test within the emulation framework (Rhizome)

2.3 Users/curators test and evaluate developed tools and concepts (Rhizome/Yale)

WP.3 Implementation

3.1 Bridge the gap in between software archive / and emulation framework (Freiburg)

3.2 Build a workflow to ingest external software packages to be used with the emulation framework (Freiburg)

3.3 Build workflow to create emulation environments from software packages

3.4 Implement extended characterization of artifacts and collections (Freiburg)

3.5 Publish technical documentation and source code (Freiburg / Rhizome)

WP.4 Management, Maintenance and Sustainability

4.1 Prototypical, semi-formal sharing platform for metadata and documentation (Rhizome)

4.2 Workflow development and best practices guide (Rhizome/Yale)

4.3 Outreach (Rhizome/Yale)

Milestones WP.1

M1.1 Y1Q1 Software archive available and ready to be used (Rhizome / Yale)

Deliverables WP.1

D1.1 Y1Q2 SW-Archive Interface Requirements (Rhizome / Freiburg / Yale)

D1.2 Y1Q2 Technical metadata requirements (Freiburg / Rhizome)

Milestones WP.2

M2.1 Y1QX Intermediate software collection for test and evaluation purposes (Yale)

M2.2 Y1Q4 Intermediate evaluation report 1 (Rhizome / Yale)

M2.3 Y2Q2 Intermediate evaluation report 2 (Rhizome / Yale)

Deliverables WP.2

D2.1 Y2Q4 Final software collection (Yale)

D2.2 Y1Q2 Manual characterization of artifact collection (Rhizome)

D2.3 Y2Q2 Emulated software environments rendering artifact collection (Rhizome)

D2.4 Y2Y4 Final evaluation report (Rhizome / Yale)

Deliverables WP3

D3.1 Y1Q3: Formal Interface to connect to software archives (Freiburg)

D3.2 Y2Q1: Workflow for management of software components (Freiburg)
D3.3 Y2Q2: Workflow for management of software environments (Freiburg)
D3.4 Y2Q3: Design and Implementation of Extended Object-Characterization (Freiburg)
D3.5 Y2Q4: Technical documentation (Freiburg / Rhizome)
D3.6 Y2Q4: Project's source code published on github (Freiburg / Rhizome)

Milestones WP4
M4.1 Y2Q1: Platform for information sharing (Rhizome)
M4.3 Y2Q3/4: Outreach (Rhizome / Yale)
Deliverables WP4
D4.2 Y2Q4: Best practices guide (Rhizome / Yale)

V. Communications Plan

As this project endeavors to ease adoption of emulation technologies by libraries, direct communications with professionals at memory institutions will be of central concern. The project represents a deepening of an established collaboration between Rhizome, University of Freiburg, and Yale University and will therefore leverage past interest (like that from the previous NEH/DFG grant to develop citation tools in bwFLA) for increased attention. Each institution's considerable networks and communications channels, moreover, can be exploited for project visibility.

Throughout the project timeline, all software and data sets created during its activities will be released under permissive free and open source licenses, and will be distributed with a road map for institutions that want to adapt the use of emulation or offer emulation services. The project participants regularly publish and speak on their ongoing research (see key staff resumes), as well. The public release of all project developments and regular communications about research are engines for professional feedback, which can then be integrated back into the project.

Additionally, some 70 universities and libraries hold subscriptions to the Rhizome ArtBase, including Brown University, California Institute of the Arts, Harvard University, and Stanford University. Rhizome staff regularly reach out to librarians at these institutions with digital preservation updates, and this project will be integrated into those communications. Increased subscriptions will be a strong indicator of growing engagement with emulation and interest in its possibilities.

Finally, with regard to secondary audiences beyond standard professionals, the digital preservation program at Rhizome is at the core of its mission and integrated into all elements of its multi-tiered public programming, from online exhibition to publishing to events. During the project period, Rhizome will produce at least two live events at the New Museum focused on digital preservation questions that will highlight the research. The organization has significant event experience, and all events sell out (165 seats) and are later archived for online distribution (with thousands of views). Additionally, Rhizome will publish two general-audience articles related to the research on rhizome.org, reaching an online public of more than 1.7 million.

VI. Sustainability

In addition to the positive effects the project will have on access to and the understanding of digital collections, and the desired lowering the barrier to adoption of emulation by libraries and other memory

institutions (as described in "Impact"), this project can serve as leverage for the long-term interests of the broader digital preservation community.

The decentralized technical infrastructure inherent to the project can serve as a basis for inter-lending performing software and combining software from different archives. This can bolster the development of new software business models, around general use or domain-specific software and knowledge about it.

Secondly, the new model of software archive created for this project with freely available software can provide a point of departure for negotiations with license holders. This might yield new pay-per-use cost models or serve as the foundation for an organization collecting and distributing royalties for legacy software ("software performance rights" akin to music performance rights).

Finally, by bridging the gap between software archives and emulation frameworks, and allowing domain experts to contribute their knowledge about software, Emulation as a Service will getting closer to becoming a sustainable service model that can prove useful to a wide audience of memory institutions, corporations, and individuals.

Abbreviated Work Plan[1]

	Y1				Y2			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
WP1	M1.1	D1.1 D1.2						
WP2		D2.1		M2.2		M2.3 D2.3		D2.1 D2.4
WP3			D3.1		D3.2	D3.3	D3.4	D3.5 D3.6
WP4					M4.1			D4.2 M4.3

WP.1 Concepts and Infrastructure Planning

Provide basic infrastructure—set up example software archive ("the software archive") and emulation framework. A detailed technical gap-analysis will be carried out, resulting in requirements for access to the software archive and interaction between the software archive and emulation framework.

Milestones WP.1
M1.1 Y1Q1 Software archive available and ready to be used (Rhizome / Yale)

Deliverables WP.1
D1.1 Y1Q2 SW-Archive interface requirements (Rhizome / Freiburg / Yale)
D1.2 Y1Q2 Technical metadata requirements (Freiburg / Rhizome)

WP.2 Prepare Use-Case, Test & Evaluation

Connect conceptual and technical development with an ambitious use-case. This work package spans the whole project period and will provide frequent feedback from users to guide development. Tools created will provide feedback on missing software components or alternative access or rendering options of selected artifacts, which then need to be tried and evaluated.

Milestones WP.2
M2.1 Y1QX Intermediate software collection for test and evaluation purposes (Yale)

[1] A detailed work plan can be found in the supporting documents section.

M2.2 Y1Q4 Intermediate evaluation report 1 (Rhizome / Yale)
M2.3 Y2Q2 Intermediate evaluation report 2 (Rhizome / Yale)

Deliverables WP.2
D2.1 Y2Q4 Final software collection (Yale)
D2.2 Y1Q2 Manual characterization of artifact collection (Rhizome)
D2.3 Y2Q2 Emulated software environments rendering artifact collection (Rhizome)
D2.4 Y2Y4 Final evaluation report (Rhizome / Yale)

WP.3 Implementation

Any implementation activity will be carried out on the instances/products chosen in Task 1.1 and will act as simple reference / proof-of-concept implementation for concepts and workflows developed within this project.

Deliverables WP.3
D3.1 Y1Q3: Formal Interface to connect to software archives (Freiburg)
D3.2 Y2Q1: Workflow for management of software components (Freiburg)
D3.3 Y2Q2: Workflow for management of software environments (Freiburg)
D3.4 Y2Q3: Design and implementation of Extended Object-Characterization (Freiburg)
D3.5 Y2Q4: Technical documentation (Freiburg / Rhizome)
D3.6 Y2Q4: Project's source code published on GitHub (Freiburg / Rhizome)

WP.4: Management, Maintenance, and Sustainability

Involve a wider community, during the project's timeline and beyond. We argue that only through usage can software and digital artifacts be preserved in a meaningful way, i.e. making available knowledge on re-enactment, usage, and expected performance. Hence, it is equally important to support creation and sharing of such knowledge as well as education and training of new users.

Milestones WP.4
M4.1 Y2Q1: Platform for information sharing (Rhizome)
M4.3 Y2Q3/4: Outreach (Rhizome / Yale)

Deliverables WP.4
D4.2 Y2Q4: Best practices guide (Rhizome / Yale)

Detailed Work Plan

	Y1				Y2			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
WP1	M1.1	D1.1 D1.2						
WP2		D2.1		M2.2		M2.3 D2.3		D2.1 D2.4
WP3			D3.1		D3.2	D3.3	D3.4	D3.5 D3.6
WP4					M4.1			D4.2 M4.3

WP.1 Concepts and Infrastructure Planning

Provide basic infrastructure—set up example software archive ("the software archive") and emulation framework. A detailed technical gap-analysis will be carried out, resulting in requirements for access to the software archive and interaction between the software archive and emulation framework.

Task 1.1: Setup basic infrastructure (Rhizome / Yale)
Set up local infrastructure for bit-stream preservation of software components and maintenance of associated descriptive metadata and a reference emulation framework. Based on an environmental scan, an archive solution is chosen and installed. bwFLA will be used as the emulation framework. Result: functional software archive instance and an emulation framework ready for use **(M1.1)**.

Task 1.2: Technical access requirements: SW-Archive interface (Rhizome / Freiburg / Yale)
Gather and define requirements based on the technical infrastructure from Task 1.1. Detailed conceptual or technical requirements have to be elaborated to allow technical interaction between the emulation framework and software archive. The emulation framework shall access descriptive metadata as well as a collection of individual files of a software component (e.g. ISO images). Result: a set of functional requirements **(D1.1)** to be implemented in WP.2.

Task 1.3: Technical Metadata: Connecting Software and Emulation (Freiburg / Rhizome)
Analyze and define additional metadata requirements necessary to make software usable with emulation, in particular, describing and packaging software such that it can integrated with emulation workflows, e.g. making use of persistent identifiers and interfaces defined in Task 1.2. Additional meta-data should

support the characterization process by describing a software component's capabilities, e.g. its native rendering formats, import or export formats. Result: requirements for additional metadata and their supporting tools **(D.1.2)**.

Milestones WP.1
M1.1 Y1Q1 Software archive available and ready to be used (Rhizome / Yale)
Deliverables WP.1
D1.1 Y1Q2 SW-Archive Interface Requirements (Rhizome / Freiburg / Yale)
D1.2 Y1Q2 Technical metadata requirements (Freiburg / Rhizome)

WP.2 Prepare Use-Case, Test & Evaluation

Connect conceptual and technical development with an ambitious use-case. This work package spans the whole project period and will provide frequent feedback from users to guide development. Tools created will provide feedback on missing software components or alternative access or rendering options of selected artifacts, which then need to be tried and evaluated.

Task 2.1: Prepare Use-Case Software Collection (Yale)
Provide test data and artifacts for development and evaluation of the technical infrastructure based on a specific use-case. Conduct a systematic ingest of necessary of software components for the intended use case. Result: populated software archive. Initially the collection is based on an experienced curator's knowledge **(M2.1)**. Throughout the project the software collection will be refined with tool support **(D.2.1)**, in combination with Task 2.2, by using the software archive to build execution environments.

Task 2.2: Prepare Object Collection (Rhizome)
Prepare the artifact collection serving as a test- and use-case within the emulation framework. Artifacts are characterized and assessed manually, i.e. by describing performance expectations and defining a desired software combination to re-enact the artifact **(D2.2)**. Emulated software environments, as defined in D2.2 are built and tested. Part of this process is input for the software archive *(*D.2.1*)*, in particular additional metadata, automatically generated through usage and user feedback. Result: a set of emulated software environments working with the artifact collection **(D2.3)**.

Task 2.3: Test and Evaluation (Rhizome / Yale)
Users / curators test and evaluate the tools and concepts developed. Performance and runtime information is added to software-packages and respectively the environments they are deployed in. Performance measures and overall evaluation will be based on D2.2, e.g. are performance expectations met by the chosen emulated environment?, are there other/better options?)

This task will produce at least two intermediate reports (**M2.2**. and **M2.3**) as well as a final evaluation report on the technical and conceptual outcome **(D2.4)**, ideally to be published as a conference or workshop paper.

Milestones WP.2
M2.1 Y1QX Intermediate software collection for test and evaluation purposes (Yale)
M2.2 Y1Q4 Intermediate evaluation report 1 (Rhizome / Yale)
M2.3 Y2Q2 Intermediate evaluation report 2 (Rhizome / Yale)
Deliverables WP.2
D2.1 Y2Q4 Final software collection (Yale)
D2.2 Y1Q2 Manual characterization of artifact collection (Rhizome)
D2.3 Y2Q2 Emulated software environments rendering artifact collection (Rhizome)
D2.4 Y2Y4 Final evaluation report (Rhizome / Yale)

WP.3 Implementation

Any implementation activity will be carried out on the instances/products chosen in Task 1.1 and will act as simple reference / proof-of-concept implementation for concepts and workflows developed within this project.

Task 3.1 Design and Implementation of a Software Archive Facade / Interface (Freiburg)
Bridge the gap between an emulation framework and an arbitrary software archive. The goal is to design a simple interface or technical facade on side of the emulation framework to allow automated access and retrieval of individual software components from a software archive's storage or repository. The technical interface should be simple and easy to adapt, such that support for different software archives can be quickly extended.

Result: formal interface design to support different software archive back-ends **(D3.1)**. As input to this task, requirements developed in Task 1.2. *(*D1.1*)* are used. A further result of this task will be a reference implementation based on the archive set up in Task 1.1.

Task 3.2: Workflow for management of software components (Freiburg)
Build a structured workflow to make external software packages usable within the emulation framework. Existing technical metadata is extended to retrieve software through the interface developed in Task 3.1 (D3.1) and enriched with information about its (proven-through-usage) capabilities to be used by novel characterization methods. This workflow will be implemented in the emulation framework (**D3.2**), created/enriched metadata will be stored and published through methods chosen in Task 4.1.

Task 3.3: Workflow for management of software environments (Freiburg)
Aim of this task to create a workflow to build software environments using software packages prepared and annotated using the workflow from Task 3.2. Any addition (i.e. installation) of a software package to an environment will create or modify additional metadata about its build/construction process. This information will be used to assign an existing, appropriate environment to an artifact, but also to allow a re-production of the environment, e.g. in a different setting. Outcome of this task is a structured workflow, integrated within the emulation framework (**D3.3**). Resulting metadata and environments will be published automatically through methods chosen in Task 4.1.

Task 3.4: Design and Implementation of Extended Artifact Characterization (Freiburg)
Aim of this task is to design and implement tool and workflow support to characterize multi-file and multi-format objects/artifacts. The process will be based on insights gained about the object collection (D2.2).

Result of this task will be workflows and tools, capable of either identifying software dependencies or already available emulated software environments for a given artifact or collection. **(D3.4)**

Task 3.5: Technical Documentation and Code Publication (Freiburg / Rhizome)
Provide technical documentation **(D3.5)** on tools and concepts developed during this project to enable integration into other frameworks and re-implementation. Any code generated within this project will be publish in a timely manner on the project's GitHub account **(D3.6)**.

Deliverables WP3
D3.1 Y1Q3: Formal Interface to connect to software archives (Freiburg)
D3.2 Y2Q1: Workflow for management of software components (Freiburg)
D3.3 Y2Q2: Workflow for management of software environments (Freiburg)
D3.4 Y2Q3: Design and Implementation of Extended Object-Characterization (Freiburg)
D3.5 Y2Q4: Technical documentation (Freiburg / Rhizome)
D3.6 Y2Q4: Project's source code published on GitHub (Freiburg / Rhizome)

WP.4: Management, Maintenance, and Sustainability

Involve a wider community, during the project's timeline and beyond. We argue that only through usage can software and digital artifacts be preserved in a meaningful way, i.e. making available knowledge on re-enactment, usage, and expected performance. Hence, it is equally important to support creation and sharing of such knowledge as well as education and training of new users.

Task 4.1 Concept / Platform to Publish and Share Environments as well as Information on Software Capabilities (Rhizome)
It is of great importance to create a diverse software preservation community, both to ensure wide coverage of software as well as expert knowledge on their usage. Evaluate options for publication, sharing and maintaining technical metadata gathered throughout this project, information about suitable (emulated) software environments, and user feedback; initially a semi-formal platform using GIT for tracking metadata as well as providing direct, functional access from technical components, and a wiki for publishing and maintaining documentation **(M4.1)**.

Task 4.2: Workflow Development and Best Practice Guide (Rhizome / Yale)
Wrap up results from WP.1, WP.2 and WP.3 to create and publish focused best practices guides. Besides the technical documentation *(D3.5)*, there is a need for problem-oriented, practical documentation, guiding and explaining relevant emulation-related tasks. Outcome of this task is a step-by-step guide/process/workflow, explaining the necessary steps from a static archived artifact to a usable rendered artifact using emulation **(D4.2)**.

Task 4.3 Outreach (Rhizome / Yale)
Project communication and acquisition of external content partners (both objects and software) for testing, feedback, and, potentially, adoption of the proposed concepts. As a result of this work package a list of potential audience and institutions is compiled (M4.3), and approached during the last half year of the project.

Milestones WP4
M4.1 Y2Q1: Platform for information sharing (Rhizome)
M4.3 Y2Q3/4: Outreach (Rhizome / Yale)
Deliverables WP4
D4.2 Y2Q4: Best practices guide (Rhizome / Yale)

DIGITAL STEWARDSHIP SUPPLEMENTARY INFORMATION FORM

Introduction
The Institute of Museum and Library Services (IMLS) is committed to expanding public access to federally funded research, data, software, and other digital products. The assets you create with IMLS funding require careful stewardship to protect and enhance their value, and they should be freely and readily available for use and re-use by libraries, archives, museums, and the public. However, applying these principles to the development and management of digital products is not always straightforward. Because technology is dynamic and because we do not want to inhibit innovation, we do not want to prescribe set standards and best practices that could become quickly outdated. Instead, we ask that you answer a series of questions that address specific aspects of creating and managing digital assets. Your answers will be used by IMLS staff and by expert peer reviewers to evaluate your application, and they will be important in determining whether your project will be funded.

Instructions
If you propose to create any type of digital product as part of your project, complete this form. We define digital products very broadly. If you are developing anything through the use of information technology (e.g., digital collections, web resources, metadata, software, or data), you should complete this form.

Please indicate which of the following digital products you will create or collect during your project
(Check all that apply):

	Every proposal creating a digital product should complete …	Part I
	If your project will create or collect …	**Then you should complete …**
☐	Digital content	Part II
✔	Software (systems, tools, apps, etc.)	Part III
✔	Dataset	Part IV

PART I.

A. Intellectual Property Rights and Permissions

We expect applicants to make federally funded work products widely available and usable through strategies such as publishing in open-access journals, depositing works in institutional or discipline-based repositories, and using non-restrictive licenses such as a Creative Commons license.

A.1 What will be the intellectual property status of the content, software, or datasets you intend to create? Who will hold the copyright? Will you assign a Creative Commons license (http://us.creativecommons.org) to the content? If so, which license will it be? If it is software, what open source license will you use (e.g., BSD, GNU, MIT)? Explain and justify your licensing selections.

The copyright for software will be held by the code authors. The source code for every piece of software produced in the course of the project will be released under the MIT license. This very permissive license has been chosen as the project's software aims to close a gap between various systems and needs to be freely adaptable to different contexts.

The dataset created during the project will be highly dynamic and will also be subject to the MIT license. The dataset is more of a set of instructions than a database, this is why no special database license was chosen.

A.2 What ownership rights will your organization assert over the new digital content, software, or datasets and what conditions will you impose on access and use? Explain any terms of access and conditions of use, why they are justifiable, and how you will notify potential users about relevant terms or conditions.

The MIT license only demands that a copyright and permission notice shall be included in all copies or substantial portions of the software. No other restrictions exist.

A.3 Will you create any content or products which may involve privacy concerns, require obtaining permissions or rights, or raise any cultural sensitivities? If so, please describe the issues and how you plan to address them.

No such content or product will be produced in the course of the project.

Part II: Projects Creating or Collecting Digital Content

A. Creating New Digital Content

A.1 Describe the digital content you will create and/or collect, the quantities of each type, and format you will use.

A.2 List the equipment, software, and supplies that you will use to create the content or the name of the service provider who will perform the work.

A.3 List all the digital file formats (e.g., XML, TIFF, MPEG) you plan to create, along with the relevant information on the appropriate quality standards (e.g., resolution, sampling rate, or pixel dimensions).

B. Digital Workflow and Asset Maintenance/Preservation

B.1 Describe your quality control plan (i.e., how you will monitor and evaluate your workflow and products).

B.2 Describe your plan for preserving and maintaining digital assets during and after the award period of performance (e.g., storage systems, shared repositories, technical documentation, migration planning, commitment of organizational funding for these purposes). Please note: You may charge the Federal award before closeout for the costs of publication or sharing of research results if the costs are not incurred during the period of performance of the Federal award. (See 2 CFR 200.461).

C. Metadata

C.1 Describe how you will produce metadata (e.g., technical, descriptive, administrative, or preservation). Specify which standards you will use for the metadata structure (e.g., MARC, Dublin Core, Encoded Archival Description, PBCore, or PREMIS) and metadata content (e.g., thesauri).

C.2 Explain your strategy for preserving and maintaining metadata created and/or collected during and after the award period of performance.

C.3 Explain what metadata sharing and/or other strategies you will use to facilitate widespread discovery and use of digital content created during your project (e.g., an API (Application Programming Interface), contributions to the Digital Public Library of America (DPLA) or other digital platform, or other support to allow batch queries and retrieval of metadata).

D. Access and Use

D.1 Describe how you will make the digital content available to the public. Include details such as the delivery strategy (e.g., openly available online, available to specified audiences) and underlying hardware/software platforms and infrastructure (e.g., specific digital repository software or leased services, accessibility via standard web browsers, requirements for special software tools in order to use the content).

D.2 Provide the name and URL(s) (Uniform Resource Locator) for any examples of previous digital collections or content your organization has created.

Part III. Projects Creating Software (systems, tools, apps, etc.)

A. General Information

A.1 Describe the software you intend to create, including a summary of the major functions it will perform and the intended primary audience(s) this software will serve.

A first tool connecting emulation frameworks with existing software archives, including metadata about the performance and capabilities of the archived software. The tool is meant to serve as a facade for different types of archives that do bitstream preservation. The software is of use to collection managers and "software curators" at libraries and archives, and will serve as a reference implementation to combine software archives and emulation frameworks.

A second tool that is a workflow for collecting metadata about the performance and capabilities of software pieces by performing them within an emulation framework. The software is of use to collection managers and "software curators" at libraries and archives, and will serve as a reference implementation for collecting performance and capability metadata on software.

Finally, a service to characterize collections of digital artifacts and match them with an appropriate emulated environment to be performed within an emulation framework, based on the metadata collected and managed with the first two tools. The software is of use collection managers and "software curators" at libraries and archives.

A.2 List other existing software that wholly or partially perform the same functions, and explain how the tool or system you will create is different.

The proposed tool's functions are novel, but involve the integration of existing tools and earlier work. Specifically:
- PRONOM (DROID or Siegfried) for characterization
- bwFLA, emulation framework created by the project partner University of Freiburg
- Bitstream-preservation archival tool, exact tool to be determined during the project

B. Technical Information

B.1 List the programming languages, platforms, software, or other applications you will use to create your software (systems, tools, apps, etc.) and explain why you chose them.

- Linux—The project partners believe that preservation projects must run on open source systems.
- Java—This is an open source language that integrates well with previous work; the emulation framework bwFLA is written in Java, and the project partners have experience with it. Since the project aims to create reference implementations, a language like Java, which has been ranked among the top 3 most popular languages for years in several indexes, will increase the readability of the code for a large audience.
- The software archive proposed as part of this project is strictly based on open source (free/libre) software.

B.2 Describe how the intended software will extend or interoperate with other existing software.

The project tool will integrate and elaborate the bwFLA emulation framework and a yet to be determined repository, but it is designed to be agnostic on both ends and run its matching service without either of these present.

B.3 Describe any underlying additional software or system dependencies necessary to run the new software you will create.

Linux and Java.

B.4 Describe the processes you will use for development documentation and for maintaining and updating technical documentation for users of the software.

We have established workflows based on Redmine and git. Documentation and publishing is assigned a time budget in the project plan.

B.5 Provide the name and URL(s) for examples of any previous software tools or systems your organization has created.

bwFLA: https://github.com/eaas-framework; http://bw-fla.uni-freiburg.de/
webrecorder: https://github.com/webrecorder/

C. Access and Use

C.1 We expect applicants seeking federal funds for software to develop and release these products under an open-source license to maximize access and promote reuse. What ownership rights will your organization assert over the software created, and what conditions will you impose on the access and use of this product? Identify and explain the license under which you will release source code for the software you develop (e.g., BSD, GNU, or MIT software licenses). Explain any prohibitive terms or conditions of use or access, explain why these terms or conditions are justifiable, and explain how you will notify potential users of the software or system.

The software will be released under the permissive MIT license. It is intended to connect existing components (archive and emulation framework) and therefore needs to be as open and adaptable as possible.

C.2 Describe how you will make the software and source code available to the public and/or its intended users.

The software and source code will be made available using Github, enabling it to reach the public and intended users.

C.3 Identify where you will be publicly depositing source code for the software developed:

Github

Name of publicly accessible source code repository: TBD
URL: TBD

Part IV. Projects Creating a Dataset

1. Summarize the intended purpose of this data, the type of data to be collected or generated, the method for collection or generation, the approximate dates or frequency when the data will be generated or collected, and the intended use of the data collected.

The project will collect data about the performance and capabilities of software. The data will be collected by executing software in combination with archive material, manually observing the behavior, and then automatically abstracting it. The process will be available to anybody using the tools developed in the project.

The dataset will be released at the end of the project, and is supposed to grow and diversify after project completion via the tool's use. The dataset is intended to enable a structural approach for re-enacting/performing legacy digital artifacts in emulation frameworks.

2. Does the proposed data collection or research activity require approval by any internal review panel or institutional review board (IRB)? If so, has the proposed research activity been approved? If not, what is your plan for securing approval?

No approval is required.

3. Will you collect any personally identifiable information (PII), confidential information (e.g., trade secrets), or proprietary information? If so, detail the specific steps you will take to protect such information while you prepare the data files for public release (e.g., data anonymization, data suppression PII, or synthetic data).

No such data will be collected.

4. If you will collect additional documentation such as consent agreements along with the data, describe plans for preserving the documentation and ensuring that its relationship to the collected data is maintained.

This is not required by the project.

5. What will you use to collect or generate the data? Provide details about any technical requirements or dependencies that would be necessary for understanding, retrieving, displaying, or processing the dataset(s).

The tools developed in the project will be used to collect the data.

6. What documentation (e.g., data documentation, codebooks, etc.) will you capture or create along with the dataset(s)? Where will the documentation be stored, and in what format(s)? How will you permanently associate and manage the documentation with the dataset(s) it describes?

This is to be determined during the course of the project. We intend to use plain-text formats like JSON or XML.

7. What is the plan for archiving, managing, and disseminating data after the completion of the award-funded project?

The dataset is highly dynamic and will be released under a free license. All participating institutions are planning to use the data in daily operations and will continue to modify and refine it.

8. Identify where you will be publicly depositing dataset(s): Github

Name of repository: TBD
URL: TBD

9. When and how frequently will you review this data management plan? How will the implementation be monitored?

The dataset will be used for daily operations after the completion of the project; it is not static and will be subject to constant refinement. The dataset is needed to generate practical results by aiding and automating the creation of emulation environments and re-enacting legacy digital artifacts. To that end, the set needs to be kept up-to-date as long as it is in use.

At the moment it does not seem necessary to keep a version history of the dataset, however the chosen publishing platform, Github, provides this functionality.

Original Preliminary Proposal

A Re-enactment Tool For Collections of Digital Artifacts

To preserve digital artifacts is to recall performances at will. Even data that is made to appear like a stable object—for instance a Microsoft Office document that resembles sheets of paper with text—depends on computer performance: without the right software and execution environment to handle the data, it will never become a "document" that is understandable to human users.

Rhizome, in close partnership with the Yale University Library and the University of Freiburg, Germany, aims to develop and release a free and open source software tool and dataset that will automatically connect collections of digital artifacts (including software) with emulation environments for re-enactment of the collections' contents. The tool will work by characterizing collections based on the well-established PRONOM[1] database and tools like DROID or Siegfried[2], connecting them with new information about software and their dependencies. The tool will either directly scan collections by itself or use existing PRONOM information provided by collection maintainers. Furthermore, we will collect and describe software required for web archives, develop an ingest and description process tightly integrated with emulation, and put the resulting tool to productive use on the existing digital collections of Rhizome and Yale University.

For instance: a 1998 website containing interactive VRML 3D models (PRONOM fmt/94) and JPEG images for textures (PRONOM fmt/44) could be viewed in a Windows XP environment using the Cortona plugin for Internet Explorer 6 or using Netscape 4.7 with Comso Player on Windows 98, since both environments are able to work with these data types. The tool will allow archivists, librarians, and digital preservation practitioners to immediately verify and feed back on these alternatives by injecting the collection into the suggested emulation environments, using the Emulation as a Service framework.[3] If no environment can be matched, the tool will allow for the creation of a new one, listing already configured environments that come closest—in this example, there might be one that is just missing a VRML plugin. This illustrates that web archives are an ideal environment to develop and test the new tool, since all software components to view websites have always been freely available, and their diversity, interactions, and system dependencies are sufficiently complex. The project can avoid licensing issues and gain significant learnings that will also be applicable to areas with difficult licensing schemes.

The result of the project will be the software tool to characterize and re-enact the contents of digital collections, usable as a local copy and a hosted service, and the connected curated software library. All software and data sets created during the project will be released under permissive free and open source licenses, including a road map for institutions that want to adapt the use of emulation or offer emulation services. The project seeks to contribute to the establishment of an approach to documenting software, disk images, and emulator configurations with the aim of establishing a standard for use by the wider community. All participants will work in dialogue with the digital preservation community, connecting with bodies like the newly forming Software Preservation Network (SPN), as well as digital conservation practitioners working in the arts and libraries.

Connecting Software Metadata with Emulation As a Preservation Tool

With emulation, the performance of legacy software can be recalled with relative ease. A meaningful integration of emulation into collections has already been successfully demonstrated for CD-ROMs, which are relatively simple, self-containing artifacts that typically only rely on standard software components.[4] By abstracting artifacts from performance environments, only a few emulation environments based on widely used configurations—releases of popular operating systems like Windows, MacOS, OS/2—need to be maintained. They enable the performance of an almost infinite amount of artifacts which can be injected into these environments from storage; for example, a legacy CD-ROM in the form of an ISO image is automatically mounted into a running emulator, making the software contained on the CD-ROM usable again.

---

[1] See: http://apps.nationalarchives.gov.uk/PRONOM/
[2] See: http://www.itforarchivists.com/siegfried
[3] See: http://eaas.uni-freiburg.de/
[4] Dragan Espenschied, Klaus Rechert, Isgandar Valizada, Dirk Von Suchodoletz and Nick Russler, "*Large-Scale Curation and Presentation of CD-ROM Art*", iPRES 2013. p.1-8, http://purl.pt/24107/1/iPres2013_PDF/Large-Scale%20Curation%20and%20Presentation%20of%20CD-ROM%20Art.pdf

However, there is a need for a more sophisticated emulation infrastructure.[5] With complex artifacts that are not as self-contained as a CD-ROM, the required technical dependencies are becoming increasingly intricate. There is no available reference describing the components of an environment for certain types of artifacts. This is especially apparent for digital collections where several artifacts need to work in concert to produce meaning, like a client-server setup, a group of related files, or websites.

Not only the artifacts we seek to preserve, but knowledge about the capabilities and dependencies of legacy software is fading quickly. This is why technical metadata must be collected practice-based, and the workflow of editing the dynamic dataset the characterization tool is based on will be integrated into the emulation environment, immediately showing the failure or success of a software combination. Hence, every claim in the resulting dataset will be based on actually performing systems and readily available software.

Participants and work plans

This project builds on insights the participating institutions gained from working together on a bilateral NEH research project[6] and their previous, extensive experience with using emulation as a preservation tool. This project represents an important facet of the ongoing efforts to transform emulation from a high-maintenance technology into a systematic tool delivering practical results for memory institutions. In 2014, the University of Freiburg was awarded with the Digital Preservation Award from the DPC for the Emulation as a Service framework, and Rhizome was shortlisted at the same event for public-facing usage of this framework.

Rhizome: Rhizome's collection of 2000 pieces of born-digital, mostly net-based art more than provides a rich ground for applying the project. Rhizome will create a new software curator position tasked with 1) collecting the freely available software required to re-enact the pieces dependent on legacy setups; 2) designing the software tool and integrating it into the collection management. Born-digital art provides many edge-cases and challenges assumptions about digital preservation in ways that have been extremely productive for the whole preservation field.[7] The project will be using and extending Rhizome's existing EaaS emulation infrastructure. Supervisor: Dragan Espenschied.

Yale University: Working with Yale's digital collections, a new digital conservator position is to be created and tasked with 1) Acquiring software from Yale's partners or the institution's legacy hardware collection, installing the software in emulated environments and describing those; 2) Creating pre-configured and documented environments with open source software on them to be used as exemplars for sharing with the wider community and for maintaining access to content that depends on the environments. The project will be done using Yale's existing EaaS emulation infrastructure. Supervisor: Euan Cochrane.

University of Freiburg (Germany): Based on previous work with the EaaS framework, which is both in use at Rhizome and Yale, existing staff will do design and implementation work on the characterization tool, its database, description tool, emulator integration, and an API to the database. Supervisor: Klaus Rechert.

Budget

Rhizome is requesting $239,000 in IMLS funds; sub-awards of $50,000 will be granted to Yale University, $70,000 to the University of Freiburg. Rhizome will contribute $22,000 in salaries and benefits via a time commitment of 20% by project director Dragan Espenschied. Yale seeks to supplement their new position with a CLIR fellowship and contribute $60,694 in salaries and benefits; Euan Cochrane is contributing 5% over two years, adding $11,500 for a total of $72,194. The University of Freiburg will contribute $17,500 for project management and supervision time by Klaus Rechert (12.5%). From the requested funds, a shared amount of $5000 will be spent on technical infrastructure and another shared $4000 on travel costs.

---

[5] See: David S.H. Rosenthal: "*Infrastructure for Emulation*", Blog Post, September 8 2015, http://blog.dshr.org/2015/09/infrastructure-for-emulation.html

[6] "Tools and Concepts for Safeguarding and Researching Born-Digital Culture–(Re-)Defining Object Boundaries and Citation of Complex Digital Objects", granted in May 2015 by the DFG/NEH Bilateral Digital Humanities Programme

[7] See: Trevor Owens: "*Digital Art Curation Grad Seminar: Your Input Welcome*", Blog Post, August 29 2015, http://www.trevorowens.org/2015/08/digital-art-curation-grad-seminar-your-input-welcome/