SWHAP Workshop, September 14th and 15th, 2023

Mathilde Fichen, Morane Gruenpeter, Jérémy Bobbio, Sabrina Granger, Roberto Di Cosmo, Jean-François Abramatic, Isabelle Astic, Emmanuelle Bermès, Camille Françoise, Claude Gomez, et al.

SWHAP Workshop
September 14th and 15th, 2023

Proceedings - October 20th, 2023

Mathilde Fichen (Software Heritage, Inria, France)

Morane Gruenpeter (Software Heritage, Inria, France)

Jérémy Bobbio *Lunar* (Software Heritage, France)

Sabrina Granger (Software Heritage, Inria, France)

Roberto Di Cosmo (Software Heritage, Inria, France)

Jean-François Abramatic (Inria, France)

Isabelle Astic (CNAM, France)

Emmanuelle Bermès (Ecole des Chartes, France)

Camille Françoise (Institute for Sound and Vision, Netherlands)

Claude Gomez (Retired, France)

Wendy Hagenmaier (Yale University and Software Preservation Network, USA)

Gregory Miura (Université de Rennes, France)

Carlo Montangero (Retired, Italy)

Simon Phipps (OSI, UK)

Kenneth Seals-Nutt (Science Stories, USA)

The content and organization of the workshop were crafted by Mathilde Fichen, with the precious help of Morane Gruenpeter, Lunar (Jérémy Bobbio), Sabrina Granger and Roberto Di Cosmo. The workshop was facilitated by Morane Gruenpeter and Lunar. The writers of the workshop proceedings are Mathilde Fichen and Morane Gruenpeter.

With the support of unesco

Abstract: In October 2022, Software Heritage hosted its inaugural SWHAP Days [1], a two-day conference dedicated to software preservation. In 2023, the Software Heritage team decided to organize a two-day, hands-on workshop in a closed committee format, scheduled for September 14th and 15th, 2023, at the Inria Paris centre. The workshop aimed to bring together professionals from diverse backgrounds, including conservation and heritage experts, researchers, and engineers. The objective was to foster collaboration, leveraging their collective knowledge and expertise to generate tangible and valuable outcomes for the community. Two topics were selected for this workshop: (1) building a guidebook on legacy software preservation and (2) telling the stories of legacy software.

Contents

1 Executive Summary **5**

2 Workshop attendees **6**

3 Agenda **10**

4 Setting the stage **11**

5 A comprehensive guidebook on legacy software preservation **12**
5.1 Identifying the audience for a guidebook on software preservation . . . 12
5.2 Why do we need a Landmark Legacy Software Preservation guidebook 26
5.3 Identifying the guidebook key questions, sections and chapters 28
5.4 Building the chapters 33

6 Telling software stories **35**
6.1 Building the storyboard for a legacy software 35
6.2 Defining a common template 47
6.3 The Software Stories interface 49
6.4 Synthetizing our thoughts on software story telling 52

7 Conclusion and next steps **58**

List of Figures

1 Persona activity - Billie . . . 14
2 Persona activity - Haïcha . . . 17
3 Persona activity - Tanja . . . 19
4 Persona activity - John . . . 22
5 Persona activity - Pablo . . . 24
6 Why a guidebook on legacy software preservation? . . . 26
7 Table of content key questions and titles . . . 28
8 Pacman storyboard . . . 37
9 Eliza storyboard . . . 40
10 The birth of the Web storyboard . . . 42
11 Fortran storyboard . . . 45
12 Software storyboards best practices . . . 47
13 Screenshot of Amaya Software Story . . . 49
14 Feedback on Amaya story . . . 50
15 What are good angles to present a software? . . . 52
16 What to ask when interviewing a historical developer of the software? 54
17 How to present a piece of code? . . . 56
18 Looking back on the guidebook, what materials in collecting efforts? . 57
19 Result of the wrap-up session . . . 58
20 Chapter 1 - Intro . . . 61
21 Chapter 2 - Who? Uses Cases . . . 61
22 Chapter 3 - Assembling the collection (1/2) . . . 62
23 Chapter 3 - Assembling the collection (2/2) . . . 62
24 Chapter 4 - Processes (1/3) . . . 63
25 Chapter 4 - Processes (2/3) . . . 63
26 Chapter 4 - Processes (3/3) . . . 64
27 Chapter 5 - Telling the story/stories . . . 64

1 Executive Summary

On September 14th and 15th, 2023, 15 professionals with a keen interest in preserving legacy software gathered at Inria Paris, at the invitation of Software Heritage. Participants were invited to pool their expertise and knowledge to contribute to two distinct areas of interest for the software preservation community.

During the initial day of our workshop, our primary objective was to exchange ideas and outline the table of contents for a Landmark Legacy Software Preservation Guidebook. In the morning session, we diligently worked to identify and understand the potential audience for the guidebook by creating personas in small groups. This exercise provided valuable insights into the specific needs and preferences of our target readers. In the afternoon, our goal was to establish a robust structural foundation for the guidebook by crafting a detailed table of contents.

Developing this table of contents helped us recognize the necessity for both high-level explanations regarding legacy software preservation and practical, technical explanations and examples. In doing so, we aim to make our guidebook accessible to a broad audience, ranging from decision-makers in cultural institutions to practitioners.

On the second day of our workshop, our central focus was to explore approaches to presenting the narratives of legacy software in a digital format. In the morning session, we concentrated on the conceptual aspect of storytelling by creating pen-and-paper storyboards for selected historical software. This visual mapping exercise enabled us to effectively plan and organize the storytelling process, identifying best practices along the way.

In the afternoon, we examined Software Stories [2], a real-life example of digital tool for presenting legacy software. Our goal was to compare our ideal storyboard from the morning session to an actual implementation and identify potential area of improvement for Software Stories. The ultimate objective for the day was to create guidance for compelling and informative digital narratives, regardless of the chosen format and tools. Among other insights, we recognized the importance of anchoring our software narratives in elements relatable to the desired audience (such as historical context, societal issues, or other software).

These two days of intensive work facilitated the exchange of expertise and experience within the cultural heritage domain, propelling us forward in our mission to provide comprehensive guidance to the community.

2 Workshop attendees

The workshop attendees are sorted in alphabetical order.

Jean-François Abramatic (Inria, France) A senior scientist at Inria, his career has been shared between research and industry. He started as a research scientist in differential games and digital image processing. As Director of Development and Industrial Relations at Inria, he launched the European host of the World Wide Web Consortium (W3C) and then moved to MIT as W3C Chairman from 1996 to 2001. He has been Vice President of Research and Development of the software company ILOG, until after its acquisition by IBM, and then became Director of Development Productivity and Innovation in the IBM Enterprise Transformation division. In 2014, he returned to Inria in the Transfer and Innovation division. He has served on the ICANN Board and is currently a member of the governing board of the European Institute of Innovation and Technology (EIT) and a member of the Coordination Council of the NETmundial Initiative.

Isabelle Astic (CNAM, France) Computer science and digital network curator at musée des Arts et Métiers - Le Cnam. After 13 years as a developer then project manager in an IT service industry, she decided to change her career path and took a course in art history to become a tour speaker. After a Master's degree in art history, and 6 months' experience in the Regional Heritage Inventory department, she joined the Musée des Arts et Métiers as curator of IT collections and digital networks. She is responsible for studying, documenting and monitoring the sanitary condition of computerized objects. Her aim is to enrich this collection with both hardware and software and to present and explain them to the public.

Emmanuelle Bermès (Ecole des Chartes, France) Dr Emmanuelle Bermès has worked 20 years on digital projects in cultural heritage institutions, mainly the French national library. Today, she works as an assistant professor at Ecole nationale des chartes (PSL University) where she is in charge of a master's degree related to technology applied to historical documents.

Roberto Di Cosmo (Software Heritage, France) After teaching for almost a decade at Ecole Normale Supérieure in Paris, Roberto Di Cosmo became full professor in Computer Science at University Paris Diderot. He is currently on leave at Inria to lead the Software Heritage project. His research interests span a wide spectrum from foundational aspects of logical systems to functional programming, parallel and distributed programming. He created and directed the European research project Mancoosi to improve the quality of large collections of software quality, and is investigating now the scientific problems posed by the general adoption of Free Software, with a particular focus on static analysis of large software collections. A

long term Free Software advocate, contributing to its adoption since 1998, he has created the Free Software thematic group of Systematic in October 2007, which has helped fund over 40 research and development projects, and he is now director of IRILL, a research structure dedicated to Free and Open Source Software quality.

Mathilde Fichen (Software Heritage, France) Mathilde Fichen trained as an engineer at Mines Paris PSL and Corps des Mines. After working in various positions for the French administration, she decided to take an academic turn in her career and earned a master's degree in the history and philosophy of science in 2022. In her current role at Software Heritage, Mathilde Fichen works on developing and promoting the SWHAP process and the Software Stories tool. She is also about to begin a Ph.D. program in the history of computer science at the Conservatoire National des Arts et Métiers. Her passions include science communication and promoting scientific knowledge to a wide audience.

Camille Françoise (National Institute for Sound and Vision, Netherlands) Camille Françoise is Open Culture/GLAM Advocate and Project manager at the Nederlands Instituut voor Beeld en Geluid. She worked in several digital departments of museums in France and abroad. In 2021, she joined Creative Commons as the Open Culture manager. As part of those activities, she participated in international advocacy activities within decision-making bodies in favour of heritage preservation. Camille Françoise is a Software Heritage ambassador since 2023.

Claude Gomez (Retired, France) Claude Gomez was a Senior Scientist at Inria. He is co-author of books on Computer Algebra and on Scilab, the open source scientific computing software. Involved in the development of Scilab since 1990, he has headed the Scilab Research and Development team since its creation in 2003 at Inria, and he was the director of Scilab Consortium. He was one of the founders of Scilab Enterprises company, created in 2010 to provide development, support and services around Scilab. He was CEO of Scilab Enterprises from 2010 to 2015, then Advisor until 2017. He is now retired.

Sabrina Granger (Software Heritage, France) Sabrina Granger is open science community manager at Software Heritage. Her role is to strengthen collaborations and to support the development of communities of practice. She holds a PhD in Humanities. For several years, she had been organizing and delivering training sessions on open science for researchers. She also led projects such as writing from scratch a whole handbook on reproducible research, in a couple of days with motivated researchers or a translation sprint of the Foster Open Science Handbook. From 2018 until 2021, she was the co-chair of the "Skills and training" working group from the French Committee for Open Science.

Morane Gruenpeter (Software Heritage, France) After several years as a professional harpist, Morane embarked on a new journey in software engineering. Her transition led her to the Software Heritage team in 2017, as she pursued a Masters degree in Computer Science at University Pierre et Marie Curie. Morane's primary responsibility revolves around representing Software Heritage and the INRIA research center in several European projects. Starting from 2021, Morane has taken on the role of coordinating the SWHAP and Software Stories project, working in close collaboration with sciencestories.io and Pisa University, and with the support of UNESCO. In 2022, she orchestrated with Elisabetta Mori the inaugural SWHAP Days event at Sorbonne University, in collaboration with LIP6. Morane is deeply committed to raising awareness about the importance of software source code preservation and building bridges within the software engineering and digital preservation communities.

Wendy Hagenmaier (Yale University and Software Preservation Network, USA) Wendy Hagenmaier is the Software Preservation Program Manager at Yale University Library, where she leads the Emulation-as-a-Service Infrastructure (EaaSI) program to empower widespread use of emulation for interaction with software, computer systems, collections, and data. She also serves as Strategic Coordinator for the Software Preservation Network's Coordinating Committee and seeks to foster collaboration and build alliances among organizations engaged in software preservation and curation. In 2022, she co-authored a white paper on "Supporting Software Preservation Services in Research and Memory Organizations," based on survey and interview research with practitioners engaged in software preservation activities.

Lunar (Software Heritage, France) Lunar learnt his alphabet on a family computer hooked to a television set in the mid-eighties. They have been playing with software ever since. Using their experience from Debian and the Tor Project, they have cofounded the Reproducible-Builds.org initiative to help build systems that provide a verifiable path from source to binary. After a few years of focusing on learning and teaching about the social aspects of digital issues, they have joined Software Heritage in 2022.

Gregory Miura (Université de Rennes, France) Newly appointed Deputy director of the library and cultural service of the Bretagne Occidentale University, Gregory Miura was previously University Librarian and director of the Bordeaux Montaigne University, head of the Library and Archives service. At the beginning of his carrer, he was in charge of the heritage collection of native digital documents on physical carrier at the French National Library (BnF) from 2002 to 2007. He is also the

chairman of the International Standards Organisation, technical committee 46 "Information and Documentation", as well as member of the source codes and software working group of the National Open Science Committee. Active in the heritage, higher education and research fields for more than 20 years, Gregory Miura was a member of the scientific advisory board of the Higher education and Research, bibliographic national agency (ABES) between 2013 and 2022.

Carlo Montangero (Retired, Italy) has been professor of Computer Science at the University of Pisa from 1981 to 2014, when he retired. There, he first taught coding -with a formal approach- and later software engineering. His research interests evolved from programming languages and methods, tools, and environments to Software Process Modelling and Enactment. At the time of retirement, he had developed an interest for the history of computing, and it was then natural that, after attending a seminar by Di Cosmo in Pisa in 2018, he became involved with Software Heritage, partecipating in the development of the SWHAP Guide [3].

Simon Phipps (OSI, UK) Simon Phipps has played many roles during a career in software and networking reaching back to the 1980s, including as part of the team that introduced Java to IBM in the 1990s and as head of open source at Sun Microsystems in the 2000s where he supervised open sourcing Sun's extensive software portfolio (Solaris, Java and more). For the last 15 years he has focussed on the Open Source Initiative (OSI) where as president he led its transformation beyond licensing into community-based software freedom and where he now works as director of policy and standards. He became involved with Software Heritage as a co-author of the Paris Call [4] and believes open source is at the foundation of civil rights and personal liberty in the digital age. Simon Phipps is a Software Heritage ambassador since 2023. See more via webm.ink

Kenneth Seals-Nutt (Science Stories, USA) Kenneth Seals-Nutt is a software engineering leader and technologist with a background in industry and academia. His work explores connections between computer science and cultural heritage, using the semantic web to bridge the gender and cultural gaps in the sciences. He has technical research experience in the digital humanities, preservation, and linked data. Along with Kat Thornton, Kenneth is Co-Founder of Science Stories, a research collective dedicated to raising the online presence of under-represented heroes in STEM. He is also in Software Engineering Leadership at Wizard, an AI conversational commerce company, Software Engineer for WikiDP and EaaSI at Yale University Library, former Director of Software Engineering at Verb Energy, and former Senior Software Engineer at SiriusXM-Pandora. Through his collaboration projects, Seals-Nutt develops interoperable knowledge-bases that promote open access to structured metadata with visual user interfaces, such as developing

the Software Stories [5] web application with Software Heritage.

3 Agenda

Day 1	Drafting a comprehensive guidebook on software preservation
9:00-9:30	Welcome coffee
9:30-10:00	Workshop launch
10:00-13:00	Identifying the audience for a guidebook on software preservation
13:00-14:30	Lunch break
14:30-18:00	Building a detailed table of content for a guidebook on software preservation
19:00-21:00	Dinner

Day 2	Presenting the history of legacy software
9:00-9:30	Welcome coffee
9:30-12:30	Building the storyboard for a software story
12:30-14:00	Lunch break
14:00-16:30	The future of Software Stories
16:30-17:30	Wrapping up

4 Setting the stage

The workshop started with a short introduction [6] by Roberto Di Cosmo, Software Heritage founder.

Software Heritage was created in 2018 on the idea that Source Code is a precious source of knowledge [7]. The Apollo 11 source code or the Quake III source code are two codes archived on Software Heritage and serve as good example of the richness that can be found in the comments left by their developers. As Len Shustek from the Computer History Museum puts it *"Source code provides a view into the mind of the designer"*. In 2019 UNESCO, Inria and Software Heritage published the Paris Call on Software Source Code [4], drawing attention of the international community of the importance and urgency of preserving software source code as a culture artefact. *"[We call to] support efforts to gather and preserve the artifacts and narratives of the history of computing, while the earlier creators are still alive."*

Software Heritage is now the largest archive for software source code, counting billions of archived source files. The initiative relies on transparency and openness and sustainability. Not a single line of proprietary software is used to build the archive, thus guaratying the sustainability of the archive in time.

Regarding the preservation of legacy software, the Software Heritage Acquisition Process (SWHAP) [3] was built in 2019 in partnership with Pisa University and UNESCO to allow the curation an archival of legacy source code in Software Heritage. In 2021 the scope of the SWHAP process was extended to documents, media and oral histories. A partnership with the Science Stories initiative [8] was set up to create the Software Stories [2] platform, to help present the collected materials.

Through SWHAP and Software Stories, Software Heritage humbly contributes to the vast preservation landscape. The Software Heritage team hopes that this two-day workshop will help bring the software preservation community together, share best practice across institutions and create useful content to support software preservation efforts.

5 A comprehensive guidebook on legacy software preservation

A major scientific and technical production of the 20th century, software is now part of our cultural heritage, and the question of its conservation and development emerged in the 1980s. While the interest and urgency of preserving software commons is now well recognized by heritage associations and institutions, putting such an archiving, conservation and promotion approach into practice is no less complex. We believe that the heritage community would benefit from a comprehensive guidebook on the matter of legacy software preservation, achieving the following goals:

- Centralizing existing resources and serving as entry point into the matter
- Setting the standards and best practices regarding software preservation
- Addressing a broad audience, from experienced practitioner to non-technical publics

5.1 Identifying the audience for a guidebook on software preservation

As a first steps towards this guidebook, we worked on better identifying our potential audience, using the personas approach. *"Personas are fictitious, specific, concrete representations of target users."* [9]

Personas are fictitious portraits of target users. These portraits are created from data collected directly from real users, articles, etc. The personas don't exist but are designed to be representative of the categories we wish to reach in priority with the guidebook.

We worked in small groups of two to three participants on a given persona. Each group was asked to answer the following question about their persona: What are their use cases for the guidebook? What do they expect from the guidebook? How would they evaluate that they are successful after implementing the guidebook recommendations? What are their fears and worries regarding using the guidebook processes? What resources and skills are available to them to implement the guidebook processes? What would they miss?

You will find bellow the five personas descriptions prepared before the workshop, along with the answers to the above questions worked out during the working session.

BILLIE, TECHNICAL EXPERT AT THE COMPUTER SCIENCE MUSEUM

General information

Billie is 41. They got interested in computers at a very early age, first coding on the family Minitel and then acquiring their first computer at age 12, saving from their pocket money. They contributes to open-source projects during their free time and is a strong proponent of free software.

Billie studied electronics and computer science and is now employed at the newly created computer science museum as a technical expert. They specialised in software preserving, ranging from video-games to office software. Their job is mainly to make digital copies of incoming software (which can arrive on various formats, from floppy disk to video game cartridges) and to make sure that can be read on emulators by visitors.

Billie is not involved in any romantic relationship but has a strong network of friends which they consider their extended family. Billie lives on their own with their cat Attila.

Values

Billie believes in the power of community and does not think one individual can achieve important things alone. They value individual freedom and autonomy and dismiss centralised authority.

Pain points and Frustrations

For now, the museum focuses on collecting executable sources, presented to the public thanks to emulation services on modern computers. Billie would like to find ways to recreate a more authentic experience of legacy software. Billie believes the museum should also be collecting the source code whenever available, which has not been the case so far.

Needs and Goals

Billie needs to build up their case to convince their hierarchy to invest more on software preservation: being able to execute legacy software on historical machines and better preserving source code. Billy is always keen on discovering new and efficient technical solutions that can make his work easier.

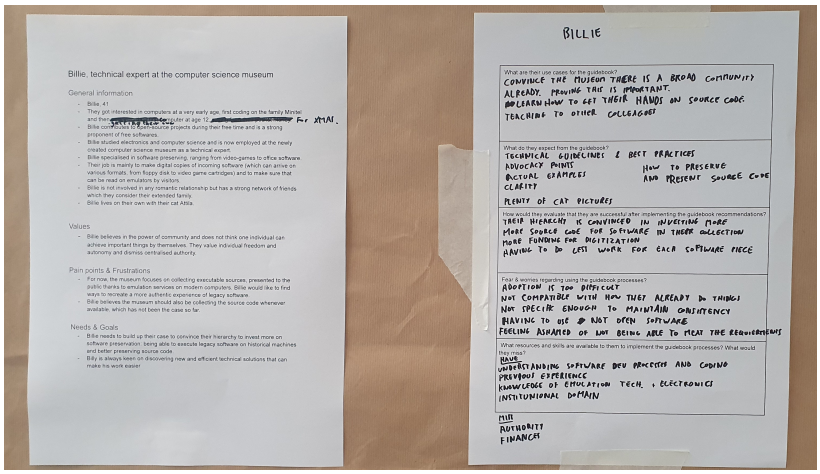

Figure 1: Persona activity - Billie

What are their use cases for the guidebook?

- Convince the museum there is a broad community already
- Proving this is important
- Learn how to get their hand on source code
- Teaching to other colleagues

What do they expect from the guidebook?

- Technical guidelines and best practices
- Advocacy points
- Actual examples
- Clarity
- How to preserve and present source code
- Plenty of cat pictures ;)

How would they evaluate that they are successful after implementing the guidebook recommendations?

- Their hierarchy is convinced in investing more
- More software source code in their collection
- More funding for digitalization

- Having to do less work for each software piece

What are their fears and worries regarding using the guidebook processes?

- Adoption is too difficult
- Not compatible with how they already do things
- Not specific enough to maintain consistency
- Having to use not open software
- Feeling ashamed of not being able to meet the requirements

What resources and skills are available to them to implement the guidebook processes? What would they miss?
Have

- Understanding of software processes and coding
- Previous experience
- Knowledge of emulation technologies and electronics
- Institutional domain

Miss

- Authority
- Finances

HAÏCHA, SCIENTIFIC CULTURE DIRECTOR AT NATIONAL INSTITUTE FOR MATHEMATICS AND COMPUTER SCIENCE

General information

Haïcha is 55. She is a former researcher in mathematics and is now working as the head of scientific culture of her research institute.

Haïcha has two teenage kids, is divorced and lives in a house in the Paris suburbs, near her work place. She sports a lot ; she jogs, and loves mountain hiking. She is more of a pen and paper person and is not so keen on using new tools or technologies.

She doesn't have a team per se except for one project manager. She relies on the researchers network to implement actions, such as science fairs, mathematics or CS tournaments, conferences etc. She has an increasing budget and the board of the institute is keen on pushing more visible and large scale scientific culture actions.

Values

Haïcha values stability, continuity, building up on the existent via incremental progress. She is very honest and straight-forward. She is suspicious of radical innovation or rapid changes.

Pain points and Frustrations The board of the institute is urging her to develop more structural actions with a larger impact. Haïcha feels like she lacks the necessary time to do so, since maintaining all the ongoing local events is already keeping her busy. The board is also keen on surfing on trendy topics, such as Big data or Artificial intelligence, which is not really Haïchas cup of tea.

Needs and Goals

Haïcha is keen on developing external partnerships to extend the reach of her projects and can invest some budget in it. The main criteria for her is the potential outreach of the projects.

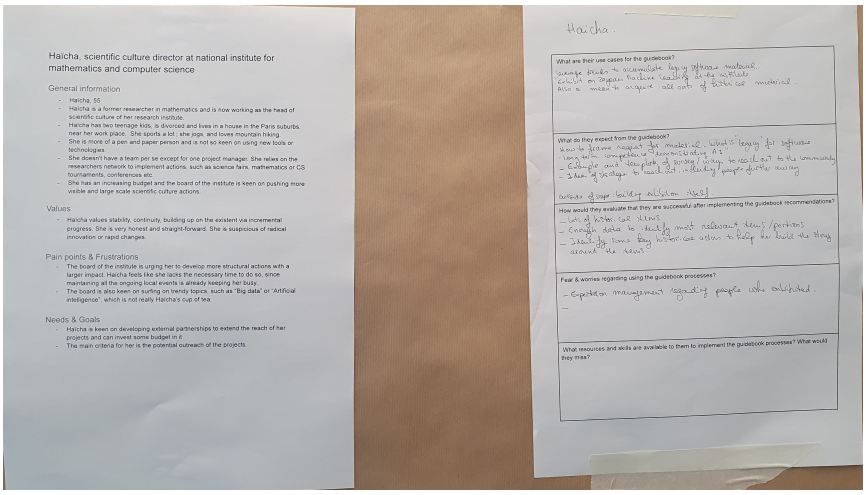

Figure 2: Persona activity - Haïcha

What are their use cases for the guidebook?

- Leverage trends to accumulate legacy software material
- Plan exhibition on "20 years of Machine learning at the institute"
- Also a means to acquire all sorts of historical material

What do they expect from the guidebook?

- How to frame a request for material (for example in a survey)
- What is "legacy" software
- Examples and templates of surveys/ways to reach out to the community
- Ideas of strategies to reach out, including to people not in direct circles
- Outside of guidebook scope: building the exhibition itself

How would they evaluate that they are successful after implementing the guidebook recommendations?

- Gather lots of historical items
- Enough data to identify most relevant items/portions
- Identify some key historical actors to help her build the story/exhibition

What are their fears and worries regarding using the guidebook processes?

- Expectation management regarding people who contributed

TANJA, SOUND, VIDEO AND MULTIMEDIA ARCHIVIST AT THE NATIONAL LIBRARY

General information

Tanja is 45. She trained at the national school for librarians and archivists. She is now an archivist at the national library, working in the multimedia department. Her role is to register incoming documents into the library catalogue, set up proper metadata and to make sure the item is properly archived in the library.

Tanja also participates in establishing the national library guidelines regarding multimedia archiving and to coordinate with multimedia editors regarding archiving their artefacts.

The bigger part of her job concerns video and audio documents, as well as video games to some extent. Other types of software are also in the scope of her department but represent a minority of her job.

Tanja is married and has an 8 years old girl. She lives in the city centre and loves cultural activities. She reads a lot (especially literature from the 19th century), goes to the museum and enjoys going to classical music concerts.

Values

Tanja is a romantic person, she values aesthetics and culture over efficiency and rationality. She will get interested in intellectual prospects but has little interest in technology.

Pain points and Frustrations

The national library does not have a clear policy regarding software archiving but is asking their staff to be more proactive in archiving software, and to reach out to the main software editors to collect artefacts. Software however is a complex object and the guidelines are not clear: which software should be archived? How to deal with always evolving versions? What object or files exactly should be archived? What metadata should be registered?

Tanja is avoiding this topic as much as she can, she feels like she didnt enrol into being an archivist to deal with such topics and that the available tools and frameworks of the library are not well suited for that.

Needs and Goals

At the moment, Tanja does not even see the point of putting efforts into software archiving. She does not see software as a cultural artefact and she feels like there is no public for software archives anyway. Getting a sense of the purpose of archiving software, from a culture and heritage perspective could help her be a bit more enthusiastic about this.

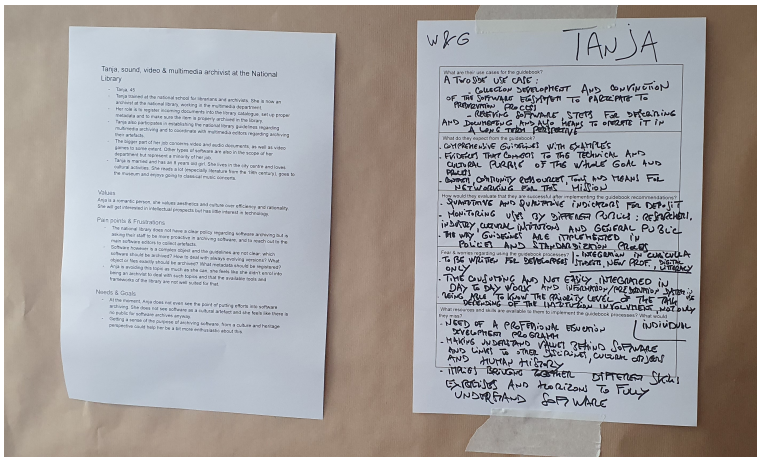

Figure 3: Persona activity - Tanja

What are their use cases for the guidebook?
A two side use case:

- Collection development and conviction of the software ecosystem to participate to preservation process
- Receiving software, steps for describing and documenting, and also means to operate it in a long term perspective

What do they expect from the guidebook?

- Comprehensive guidelines with examples
- Evidences that connect to the technical and cultural purpose of the whole goal and process
- Contacts, community resources, tools and means for networking for this mission

How would they evaluate that they are successful after implementing the guidebook recommendations?

- Quantitative and qualitative indicators for deposit
- Monitoring uses by different publics: researchers, industry, cultural institutions and general public
- The way guidelines are implemented in policies and standardization process

- Integration in curricula, students, new professions, digital literacy

What are their fears and worries regarding using the guidebook processes?

- To be written by developers only Time consuming and not easily integrated in day to day work and information/preservation systems in use
- Being able to know the priority level of the task depending on the institution involvement, not only individual

What resources and skills are available to them to implement the guidebook processes? What would they miss?

- Need of a professional education development programme
- Making understand values behind software and links to other disciplines, cultural objects and human history
- Imply bringing together different skills, expertise and horizons to fully understand software

JOHN, ACTIVE MEMBER OF THE SOFTWARE HISTORICAL SOCIETY

General information

John is 81. After a bright career as a CS engineer in various American companies, John is dedicating his retirement to preserving legacy software. He created a small non-profit called the Software Historical Society, dedicated to collecting historical software artefacts.

The SHS gathers around 5 people and has a limited budget. Because they dont have a dedicated physical storage space, they mostly focus on preserving artefacts in digital format. John is particularly proud of having recovered the source code of the first FORTRAN compiler, which can now be found on the SHS website, along with other historical material surrounding FORTRAN.

Johns wife died a couple of years ago. John has 2 children and 5 grandchildren which he loves taking care of.

Values

John is a family person. He values tradition and social order. He likes rules to be clearly defined and respected.

Pain points and Frustrations

John feels like there are so many valuable legacy software and source codes to be preserved and not enough time and resources dedicated to it. John feels like initiatives like his are scattered, badly coordinated and lacking clear guidelines.

John cant afford storing physical artefacts (like paper listings, machines, documents etc) and is afraid these will get lost forever.

Needs and Goals

John wished there were more clearly defined standards for software preservation. He would also wish there were a national digital and physical museum or archive he could directly contribute to, rather than building his own website with limited capacity.

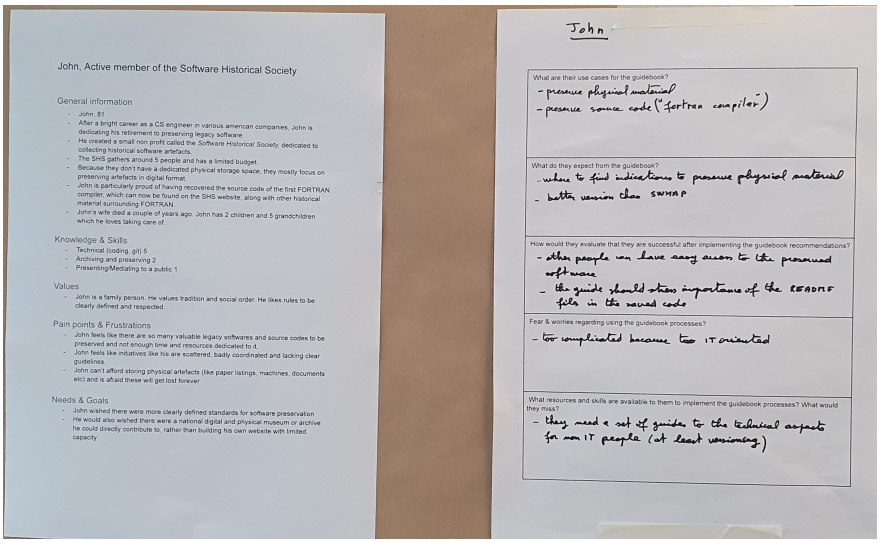

Figure 4: Persona activity - John

What are their use cases for the guidebook?

- Preserve physical material
- Preserve source code ("fortran compiler")

What do they expect from the guidebook?

- Where to find indicators to preserve physical material
- Better version than existing SWHAP guide

How would they evaluate that they are successful after implementing the guidebook recommendations?

- Other people can have easy access to the preserved software
- The guide should stress importance of the README files in the saved code

What are their fears and worries regarding using the guidebook processes?

- Too complicated because too IT oriented

What resources and skills are available to them to implement the guidebook processes? What would they miss?

- They need a set of guides to the technical aspects for non IT people (at least versioning)

PABLO, COMPUTER SCIENCE MUSEUM CURATOR

General information

Pablo is 35. After getting a bachelor in computer science and electronics he moved on to humanities and got a master degree in history and philosophy of science. He then completed a PhD where he worked on the history of the French phone industry. He works as a curator for a newly created National Museum for Computer Science. He previously occupied diverse jobs as history lecturer and freelance tech journalist. He also started a non-profit project to collect oral archives from computer science pioneers.

Pablo has an entrepreneurial mindset and likes starting new projects. He lives in the capital city of his country where he lives a busy life with his partner, going to events, exhibitions and hosting dinner parties. He doesnt have kids and doesn't want any.

This new job is a challenge for him but he feels very excited to start a museum from scratch, build the collections and find innovative ways of presenting it to the public.

Values

Pablo believes in hard work and personal merit. Success means clearly defining what your objectives are and putting in enough effort to achieve them. He is a pragmatic person.

Pain points and Frustrations

The newly created CS museum has a limited budget and choices need to be made. The museum project attracts many experts and peers interested in contributing but everyone has its own opinion on what should be done and what the priority should be.

Some people think the museum should focus on maintaining the machine in working state and running historical software on it. Some think the priority should be to collect material (machines, source code and related materials) before they are lost. Pablo struggles with establishing what the priority should be.

Needs and Goals

Preserving is important but the museum will only be a success if visitors come, which is Pablos priority. Pablo is open to using new tools but they need to be time-efficient, and give concrete results, viewable to visitors

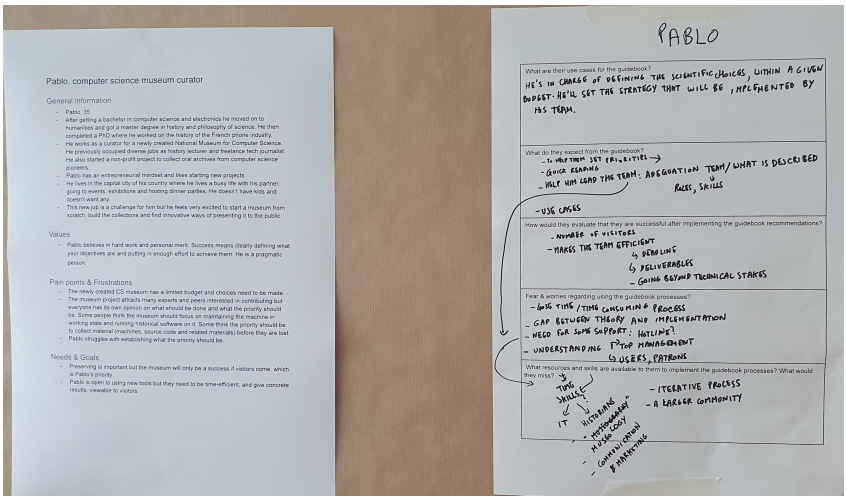

Figure 5: Persona activity - Pablo

What are their use cases for the guidebook?

- He’s in charge of defining the scientific choices, within a given budget
- He’ll set the strategy that will be implemented by his team

What do they expect from the guidebook?

- To help them set priorities
- Quick reading
- Help him lead the team: adequation between team (skills/roles) and what is described

How would they evaluate that they are successful after implementing the guidebook recommendations?

- Number of visitors
- Makes the team efficient (deadline, deliverables) beyond technical stakes

What are their fears and worries regarding using the guidebook processes?

- Lose time / time consuming process Gap between theory and implementation

- Need for some support: hotline?
- Understanding top management, users, patrons

What resources and skills are available to them to implement the guidebook processes? What would they miss?

- Time
- Skills: IT, Historians, Museology,
- Communication and Marketing
- Iterative process
- A larger community

Activity take-aways

The goal of defining personas is not to define an "average" reader, but to empathize with our audience and to make sure that the guidebook is written for real people and answers to real-life concerns. It is however possible to identify some high-level take-aways from the work done on our personas:

- The requirements for a guidebook range from very high level processes (setting up a strategy for software preservation, reaching out to contributors, convincing co-workers) to very technical items
- Specifying and showcasing concrete examples and uses cases can provide hands-on guidance
- The need for the guidebook to be flexible enough to adapt to existing processes or tools already in place
- Fear of the guidebook to be too technical, for developers only
- Decision makers should be kept in the loop to provide more suitable policies

5.2 Why do we need a Landmark Legacy Software Preservation guidebook

Taking a step back before diving into the conception of the table of content of our guidebook, we brainstormed on the need for a Landmark Legacy Software Preservation Guidebook. Each participant could freely phrase out loud their ideas which were written down on a blackboard. A picture of the result and transcript can be found below.

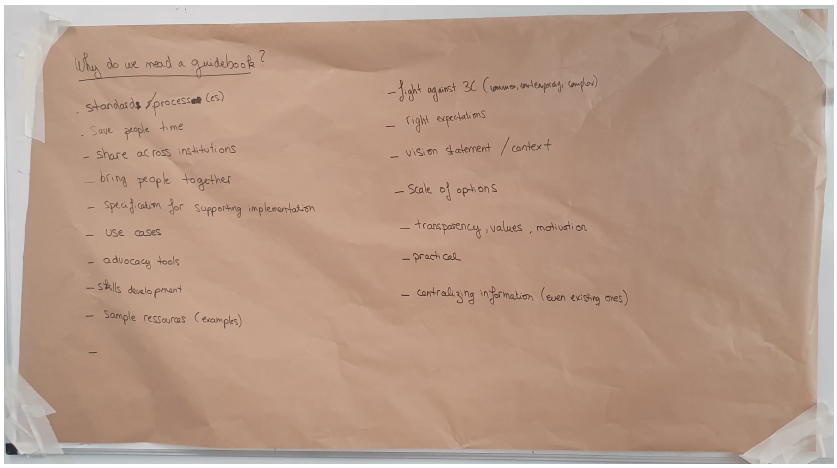

Figure 6: Why a guidebook on legacy software preservation?

Why do we need a guidebook?

- Standards / process (es)
- Save people time
- Share across institutions
- Bring people together
- Provide specification for supporting implementation
- Provide use cases
- Provide advocacy tools
- Skills developments
- Provide sample resources (examples)
- Fight against the 3C Curse (Common, Contemporary, Complex)
- Set the right expectations

- Give a vision statement / context
- Provide scale of options
- Transparency, values and motivation
- Practical
- Centralize information (even existing ones)

5.3 Identifying the guidebook key questions, sections and chapters

With our personas and main reasons for our guidebook in mind, we could start working on the structure of the guidebook itself. A first brainstorming phase aimed at identifying the questions that should be addressed by the guidebook, as well as its main sections and chapters. To do so, we shared our thoughts on two types of post-its that we could freely stick to the wall. On yellow post-its, we wrote down the main questions that the guidebook should aim to answer. Using orange post-its, we shared section titles ideas. We could move the post-its around to group questions and section titles by topics.

Once all the ideas were shared we discussed together to define five main chapters, that were written down on blue post-its. The picture of the resulting board can be seen thereafter.

Figure 7: Table of content key questions and titles

Below you will find a transcript of the board. For readability purpose we allocated the different questions and sections titles ideas into our five final chapters. Note that this allocation was not that clearly defined at that point in time, nor were the order of the chapters.

Landmark legacy software preservation guidebook

CHAPTER 1: INTRO
Section titles

- Rationale/values
- Value
- Definition(s)
- Concepts that need to be explained
- A brief history
- Software and other digital objects
- Typology of typologies
- Prepping materials
- Pre-requisites
- Clear statements on what is out of scope

Section titles

- What are the goals of this guidebook?
- What are our goals/values as an organization?
- What is the current state of software preservation?
- How does it connect to preservation in general?
- What is software?
- What a software is made of?
- What is landmark legacy software?
- Why is preserving software important?
- Why do we need to preserve software?
- Who cares?
- Where to start?
- What do I need to get started?
- What software is eligible?
- How do I know if my software is worth preserving?
- How to avoid reinforcing biases?

CHAPTER 2: WHO? USES CASES, ACTORS
Section titles

- Already contributing institutions
- Atlas of software collections
- Use cases
- Networking with other curators
- Train the trainer, teach with this book

Questions

- Who is the target audience for this book
- How do I convince my decision maker to invest resources in software preservation?
- Who uses software collections and for what?
- I have some software material, what should I do with it?
- Who has software to preserve and who can preserve it?
- Who should feel responsible for preservation software?
- How do I notify others that my institution is preservation a specific record?
- Who can help me?
- What other organization use this framework of process?
- Who are the community organizations involved in software preservation + how can I join them?
- Can I study from someone else's experience?

CHAPTER 3: ASSEMBLING THE COLLECTION
Section titles

- Copyrights and licenses (Chapter title proposal: "Better call Simon")
- Traceability (of source code?)
- Timeline
- Authors
- Attribution
- Legal/Policy by country
- Description and metadata

- Schema/Data Model for describing software
- Do's versus Don'ts

Questions

- How do I know what is important to preserve?
- What shall be preserved, for what purpose?
- What should I NOT do when trying to preserve software?
- What needs to be preserved? Eg. Source, Images, Docs, Binaries, Build, History
- Where do I keep things? Eg. Code, Media, Paperwork
- Which standard?
- Preserving associated hardware
- How can I ask authors to give a copy of the source code?
- How can I motivate people to contribute?
- How to crowd-source historical software artefacts?
- How to search for a software I know/I don't know?
- Indexation
- Comment accéder facilement à un logiciel? (How to easily access software?)
- What can we do nationaly/locally and what should we do internationally?
- Is this legal?
- Which support? Where can I find help or support?
- Where do I keep the existing metadata?
- What vocabulary to use for the metadata?

CHAPTER 4: THE PROCESS(ES) Section titles

- Clear indicator about the level of difficulty
- Level of difficulty
- Compiling some code so it can be executed and, if possible, emulated
- Preservation methods (emulation, versioning, migration...)
- Pérennité (sustainability)
- Process overview
- Cheat Sheets

- Regular recap list
- Git/VCS (Version control systems)
- Versioning
- Wikidata
- Recommendations for digital
- preservation tool builders / system maintainers for building interoperable systems that support software preservation

Questions

- Where can I see an example of a complete instance of the process?
- How to curate collected materials?
- Quel format de conservation (which preservation format)?
- Which tools?

CHAPTER 5: TELLING THE STORY/STORIES
Section titles

- Storytelling
- Presenting creators and developers
- Embedding an emulated environment for presentation
- How to do an oral history about software
- UI/UX of preserved software

Questions

- How to present a software history to an audience
- How to write a good biography?
- How to explain why some materials were selected (and others not)?
- How do I reference a piece of software in court?
- What are good questions for an interview?

OTHER
Bibliography

5.4 Building the chapters

Based on the output of this brainstorming session, five groups were created to work each on the content of a given chapters. Each group was asked to define the relevant subsections and main ideas. During the session, we could freely pick-up post-its (Questions or Section titles) that we found most relevant to our chapter, and incorporate them in our proposal. Each group then presented their output as a poster, whose picture can be found in *Appendix 1 - Guidebook chapters pictures*.

Below you will find a condensed version of the table of content we constructed based on theses posters. The full table of content can be found in *Appendix 2 - Guidebook detailed table of content*. Note that some adjustments have been made after the workshop to make the final table of content consistent (typically dealing with duplicates between two chapters, aligning subsections numbering format etc.). Some chapter titles were changed by the participants during the process.

Landmark legacy software preservation guidebook

1. Chapter 1 - Scope and Definition
    a) Overview: how to use this guide
    b) What is software?
    c) Why is it important to preserve software?
2. Chapter 2 - Use cases and existing landscape
    a) Concrete examples of software preservation initiatives
    b) The existing landscape of software preservation
3. Chapter 3 - Assembling the collection
    a) Defining your scope of software collecting
    b) What could be collected as part of your software collecting?
    c) Where can I obtain the software materials
    d) How do I collect the software?
4. Chapter 4 - The process(es)
    a) Curating software source code
    b) Curating the executables
    c) Curating physical software artifacts
5. Chapter 5 - Telling the story/stories

a) Define the purpose of your story

b) Prepare supplementary materials

c) Constructing the story

d) Publishing and promoting your story

6. Bibliography

This table of content is a first draft, and some extra work would be needed to get to a final usable version on which to build a guidebook. In particular, we did not have time after working on their dedicated section to merge everything back together, check the consistency of the whole guidebook, make sure that all the questions that came out of the initial brainstorms were handled, confront ideas and get to a consensus.

We also raised some questions and challenges that we faced, including the following:

- Should the guidebook address an institutional audience (museums, libraries etc.) or specific individuals (a curator, a librarian, a software engineer)? It is specifically important to involve current and retired software engineers, who are the ones owning the software materials.
- Do we need to give a classification of software in the first chapter? It helps to demonstrate the diversity of software, but it is also not an easy and consensual task.
- What would be the exact cut between Chapter 3 (Assembling the collection) and Chapter 4 (Processes)? Should Chapter 3 stay at a conceptual level and Chapter 4 get to concrete tools and detailed processes ("10 steps", "how to" type of information)
- Is Chapter 5 (Constructing the story) really in the scope of such a guidebook? Usually people involved with preservation are not the same than the ones involved in presenting, exhibiting the materials.
- Chapter 5 could start with the Publishing subsection: where/how you display your software story will impact the rest of the process.

6 Telling software stories

The preservation of software can be a goal in itself, independently of any pre-identified use of the collected objects. It is primarily an effort to preserve the memory for future generations, as we do not know what their questions or needs will be regarding our software heritage.

For certain institutions, like science and technology museums, legacy software preservation is however closely linked to its presentation to the public. Software is a complex object and mediating its story can be a real challenge. Some institutions engaged in such initiatives. The Computer History Museum (Mountain View, California) displays on its website a collection of blogposts about specific software histories, mixing oral stories, archives photos, legacy screenshots and textual information (see for example the SmallTalk blog post [10]).

In 2021, Software Heritage launched the Software Stories [2] initiative, in collaboration with the Science Stories team and the University of Pisa. The project leverages information available on Wikidata to present the history of significant historical software projects in an accessible and interactive way.

During the second day of the workshop, we worked on this complex topic: how to tell the story of a software to a broad and non-technical audience?

6.1 Building the storyboard for a legacy software

In this first session, we divided in small groups to build the storyboard for a given piece of legacy software. Four very different pieces of software had been selected before the workshop, ranging from a programming language to a chatbot or a video game:

- Fortran
- Eliza
- The first Web page
- Pac-Man

During this session we focused on the ideal structure for their story, and we did not worry on the actual availability of the required material. To help structure the storyboard predefined "building blocks" had been prepared beforehand, assuming the story would be presented in a digital format: picture, video, source code, timeline, portrait gallery, interactive map. These building blocks mirror to some extent those used in the Software Stories. Each team was also given some historical elements about the software in order to get started, even though we were free to make our own research.

You will find thereafter the software description given to each group, as well as a picture of the resulting storyboard along with a summary of the presentation made by the group at the end of the session.

PAC-MAN

The classic and enormously popular Pac-Man video game came out in Japan on May 21, 1980, and by October of that year it was released in the United States [11]. The yellow, pie-shaped Pac-Man character, who travels around a maze trying to eat dots and avoid four hunting ghosts, quickly became an icon of the 1980s. To this day, Pac-Man remains one of the most popular video games in history, and its innovative design has been the focus of numerous books and academic articles.

The game was created by Namco in Japan, and released in the U.S. by Midway. By 1981, approximately 250 million games of Pac-Man were being played in the U.S. each week on 100,000 Pac-Man machines. Since then, Pac-Man has been released on nearly every video game platform.

According to Japanese game designer Toru Iwatani, Pac-Man was conceived as an antidote to the overwhelming number of games with violent themes, such as Asteroids, Space Invaders, Tail Gunner, and Galaxian. Pac-Man's innovative break away from the shoot-em-up style of arcade game would crack open the video game universe.

Instead of a warrior fighting off attackers by firing ammunition at them, the Pac-Man character chews its way to victory. The game contains several references to food: Pac-Man chomps away at pills in his path, and consumes bonus items in the shape of fruits and power pellets (originally) in the shape of cookies. Each ghost's name gives a hint to its strategy for tracking down Pac-Man: Shadow ("Blinky") always chases Pac-Man, Speedy ("Pinky") tries to get ahead of him, Bashful ("Inky") uses a more complicated strategy to zero in on him, and Pokey ("Clyde") alternates between chasing him and running away. The sound effects were among the last things added to the game, created by Toshio Kai. In a design session, Iwatani noisily ate fruit and made gurgling noises to describe to Kai how he wanted the eating effect to sound.

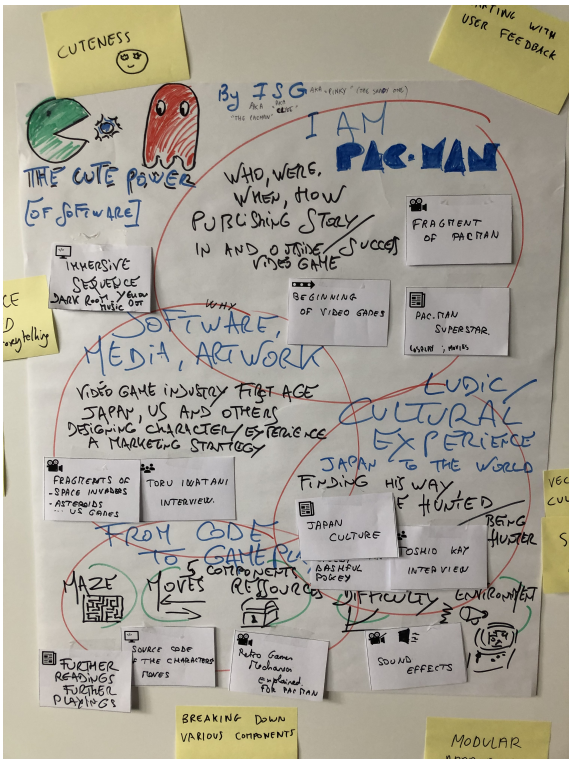

Figure 8: Pacman storyboard

The Pacman storyboard is divided in four main sections.

The first section serves as entry point to connect with the audience, emphasizing that "everyone knows Pac-Man". The section answers the Who, Where, When, How questions, showing a timeline of Pac-Man versions and how the characters became a reference inside the videogame industry, but also outside (books, movies etc.)

The second section Software, Media, Artwork, aims at telling a lesser known story of Pac-Man, relating to the broader context of the first decade of the videogame industry and the competition between Japan and the USA. Pacman was designed as a cute character in reaction to American spaceships and shoot-them-up videogames. This section would contain fragments of contemporary video-games (space-invaders, Asteroids...) and an interview of one of Pacman creator.

The third section shows Pac-Man as a ludic and culture experience. The

game can tell us about the Japanese culture (ex: the different ghosts). The section features an interview of one of Pac-Man artistic creator as well as some extracts from the sound effects.

The last section From code to game play brings the visitor behind the scene, highlighting some components of Pac-Man experience: different sets of character moves, sound effects, maze layout, difficulty levels. This section allows to explain the constraints faced by the designers, how they made their decisions, and to feature some pieces of source code.

An extra section could give some pointers towards extra resources and links to website where to play retro Pac-Man games.

ELIZA

Joseph Weizenbaum invented ELIZA in the late 1960s, the worlds first autonomous computer chat program [12]. With 200 lines of code, ELIZA was capable of holding up one end of a conversation with a human. The program was rudimentary but effectiveperhaps too effective.

The best way of achieving a conversation between a computer and a human, Weizenbaum believed, was to mimic the repetitive structure of a psychoanalysis session, with the program repeating words and rephrasing statements given to it in the form of a question. The idea, he said in 1984, was to have a program that didnt need a deep reservoir of knowledge.

A simple exchange went like this:

- User: Well my boyfriend made me come here.
- ELIZA: Your boyfriend made you come here?
- User: He says Im depressed much of the time.
- ELIZA: I am sorry to hear you are depressed.
- User: Its true. I am unhappy.
- ELIZA: Do you think coming here will help you not to be unhappy?

By the mid-1970s, Weizenbaum was sufficiently concerned about the trajectory of artificial intelligence that he authored the 1976 book *Computer Power and Human Reason: From Judgment to Calculation*, which argued computers can never be omniscient. Humans, he said, needed humans to address human problems. While this wasnt the only objection heard with regard to computers, it was one of the rare times a computer scientist was sounding the alarm.

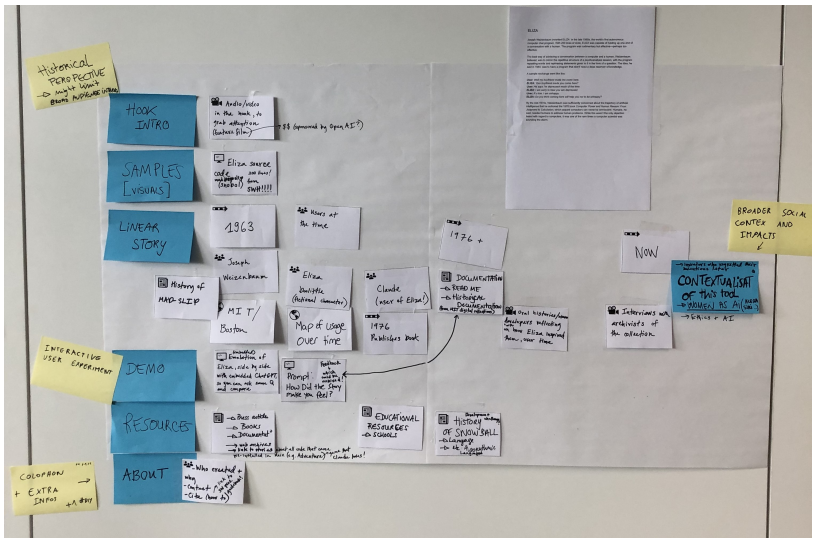

Figure 9: Eliza storyboard

The Eliza storyboard starts with a hook intro, to catch the viewer interest, relating to what we know today (typically presenting Eliza as an ancestor of ChatGPT [13]).

Eliza was a short computer program (200 lines of code), and the story would continue by showing the full code along with some chose visual samples of interactions with it.

Then, a linear story of Eliza is presented, featuring a timeline of the Eliza story, a presentation of the people and institutions involved etc. This section would feature some oral histories of the historical developers.

The story would enlarge to society themes, such as Women portrayed in AI, Ethics, Authors who regret their invention...

The viewers would be offered to interact with Eliza, alongside with ChatGPT so that they could compare the two tools. A prompt could be to ask the visitors how they enjoyed the presentation, thus allowing the curators to gather some feedback.

A Resources section would give pointers to extra material (press articles, books, educational resources). An About section explains who created the story, and why, linking to our guidebook!

THE FIRST WEB PAGE While working at CERN, Tim Berners-Lee became frustrated with the inefficiencies and difficulties posed by finding information stored on different computers [14]. On 12 March 1989, he submitted a memorandum, titled "Information Management: A Proposal", to the management at CERN. The proposal used the term "web" and was based on "a large hypertext database with typed links".

Berners-Lee's breakthrough was to marry hypertext to the Internet. In the process, he developed three essential technologies:

- a system of globally unique identifiers for resources on the Web and elsewhere, the universal document identifier (UDI), later known as uniform resource locator (URL);
- the publishing language Hypertext Markup Language (HTML);
- the Hypertext Transfer Protocol (HTTP).

With help from Robert Cailliau he published a more formal proposal on 12 November 1990 to build a "hypertext project" called World Wide Web (abbreviated "W3") as a "web" of "hypertext documents" to be viewed by "browsers" using a clientserver architecture.

By December 1990, Berners-Lee and his work team had built all the tools necessary for a working Web: the HyperText Transfer Protocol (HTTP), the HyperText Markup Language (HTML), the first web browser (named WorldWideWeb, which was also a web editor), the first web server (later known as CERN httpd) and the first web site (http://info.cern.ch) containing the first web pages that described the project itself was published on 20 December 1990.

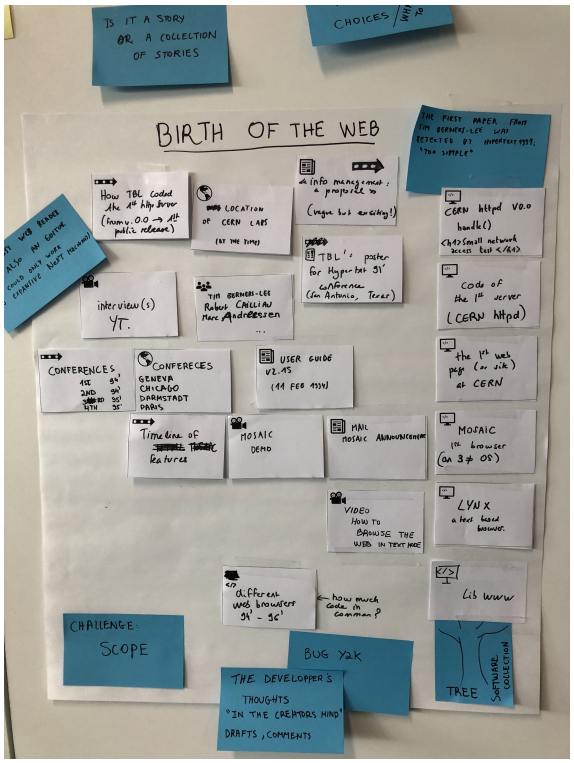

Figure 10: The birth of the Web storyboard

The group decided to rephrase the storyboard "The Birth of the Web" thus enlarging the topic beyond the first webpage itself. Since a lot has already been written and presented about the history of the Web, the group decided to focus on the different source codes which made the web history, thus choosing an more original angle.
Presented source codes would include:

- The first web pages
- The first server
- The first browser

The story would start with the historical note of Tim Berners-Lee "Information management: a proposal", from which the Web project started. The source code of the very first Web server would be presented (CERN httpd V0.0), and commented from the point of view of Berners-Lee at the time.

The code of the first web page would then be presented, alongside with historical information (timeline of the first conferences, historical user guide etc).

Because the Web source codes were distributed openly, developers could contribute to its adoption and development in a decentralized way. Many browsers were thus developed. The story would focus on the source code of Mosaic (1st browser to be distributed on many operating systems) [15]. A video would show the use of a text-based browser, navigating the web without a graphic interface !

FORTRAN

FORTRAN, in full Formula Translation, computer programming language created in 1957 by John Backus that shortened the process of programming and made computer programming more accessible [16].

The creation of FORTRAN, which debuted in 1957, marked a significant stage in the development of computer programming languages. Previous programming was written in machine (first-generation) language or assembly (second-generation) language, which required the programmer to write instructions in binary or hexadecimal arithmetic. Frustration with the arduous nature of such programming led Backus to search for a simpler, more accessible way to communicate with computers. During the three-year development stage, Backus led an eclectic team of 10 International Business Machines (IBM) employees to create a language that combined a form of English shorthand with algebraic equations.

By allowing the creation of natural-language programs that ran as efficiently as hand-coded ones, FORTRAN became the programming language of choice in the late 1950s. It was updated a number of times in the 1950s and 1960s in order to remain competitive with more contemporary programming languages. FORTRAN 77 was released in 1978, followed by FORTRAN 90 in 1991 and further updates in 1996, 2004, 2010, and 2018. However, fourth- and fifth-generation languages largely supplanted FORTRAN outside academic circles beginning in the 1970s.

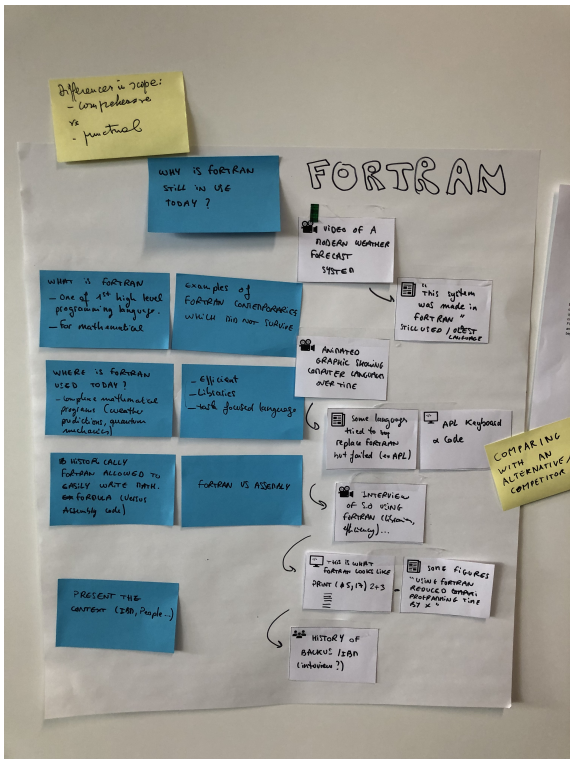

Figure 11: Fortran storyboard

The Fortran storyboard is structured around a central question that serves as a thread throughout the story: "*Despite being one of the oldest programming language, why is Fortran still in use today?*".

To help connect with the audience, the story would start with a video of a modern weather forecast system, explaining that Fortran is still in use despite being one of the oldest programming language.

An animated graphic would then show the use of the different programming languages over time, highlighting Fortran. Some languages tried to replace Fortran but failed, and the story would highlight APL, showing a code extract and a picture of an APL Keybord (programming in APL involved very weird symbols).

An interview of a Fortran expert would explain was Fortran is still a very powerful language (the numerous libraries well suited for mathematical

programs, its efficiency etc.)

Going back in time, the story would explain that before Fortran people would code in assembly languages (very time consuming) and would compard a simple operation (adding two figures for example) in Fortran and assembly.

The story would conclude by showing some historical background: history of Backus, IBM, key dates and some historical interviews.

6.2 Defining a common template

Based on the four storyboards we identified commonalities between approaches, best practices as well as things to be avoided. The ideas were recorded on a blackboard, which picture can be found hereafter.

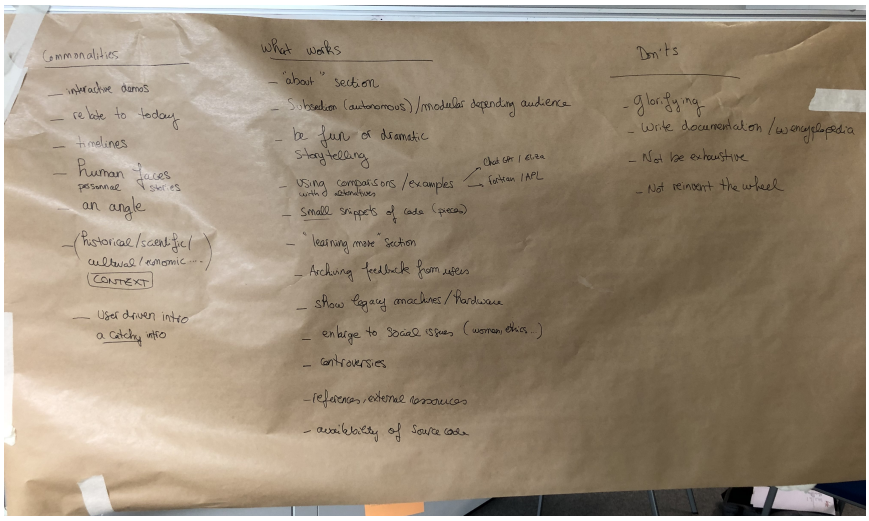

Figure 12: Software storyboards best practices

A transcript of the board can be found below.

Commonalities

1. Interactive demos
2. Relate to today
3. Timelines
4. Human faces/personal stories
5. An angle
6. Context (historical, scientific, cultural, economical...)
7. User driven intro / a catchy intro

What works

1. "About" section

2. Autonomous subsections / Modular visit depending on the audience
3. Be fun or dramatic, storytelling
4. Using comparisons with alternatives, examples (ChaptGPT vs. Eliza, Fortran vs. APL)
5. Small snippets of code
6. "Learning more" sections
7. Archiving feedback from users
8. Show legacy machines / hardware
9. Enlarge to social issues (ethics, women...)
10. Controversies
11. References, external resources
12. Availability of source code

Don’ts

1. Glorify
2. Write documentation / encyclopaedia
3. Try to be exhaustive
4. Reinvent the wheel

6.3 The Software Stories interface

After working on conceptual storyboards without any real-life constraints we turned to a concrete example: the Software Stories [2] initiative, started by Software Heritage and University of Pisa, based on a pre-existing tool developed by Science Stories.

A short presentation of Science Stories was made by Kenneth Seals-Nutt. Kenneth gave a walkthrough of the projects that led up to building Software Stories. First, he shared the Wikidata for Digital Preservation Portal (WikiDP) [17], his collaboration project with Kat Thornton and Carl Wilson for Yale University Library. Then, he discussed how Science Stories [8] was founded out of using linked data contributed to Wikidata along with IIIF [18] to celebrate women at Yale University who took part in research before undergraduate co-education was permitted. Science Stories then expanded into a visual storytelling platform for thousands of women and underrepresented scientists, prompting the development of a scaled service to support linked data-powered digital storytelling. Kenneth then demoed a Science Story [19] and showed the open source Python package [20] and JavaScript/React component library [21] that Kat and Kenneth built along with the JSON representation of the API schema. Lastly, he discussed how this framework allowed for a collaboration with Software Heritage to extend beyond biographical content into telling the story of landmark legacy software through Software Stories [5].

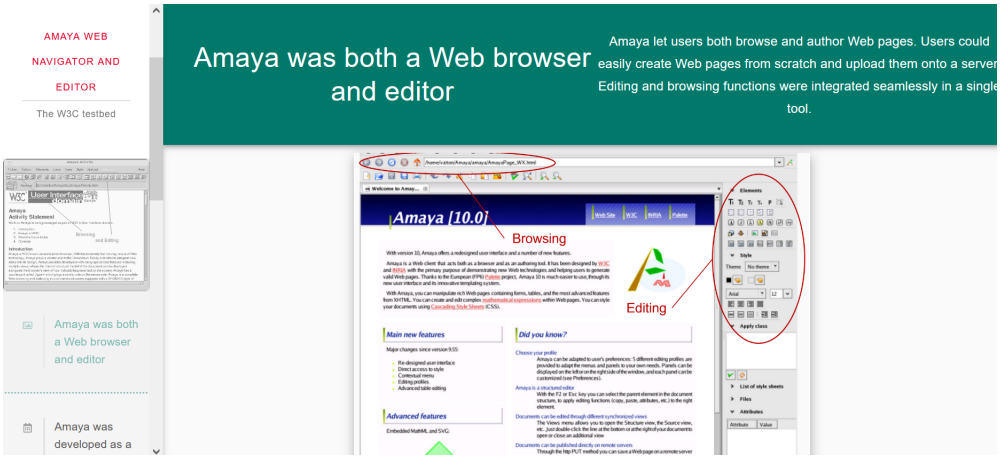

Figure 13: Screenshot of Amaya Software Story

We then took some time to navigate into a given Software Story, made about Amaya [22], a web browser and editor developed at Inria as of 1994 and adopted by the World Wide Web consortium as a testbed. We wrote down our questions and feedback regarding Amaya's software story, leveraging the best practices identified

thanks to the storyboards. The collected post-its were sticked to a board, whose picture can be found below. For readability purpose, the feedbacks are sorted in two categories in the transcript below: those relating to the tool (technical question) and those relating to the story itself.

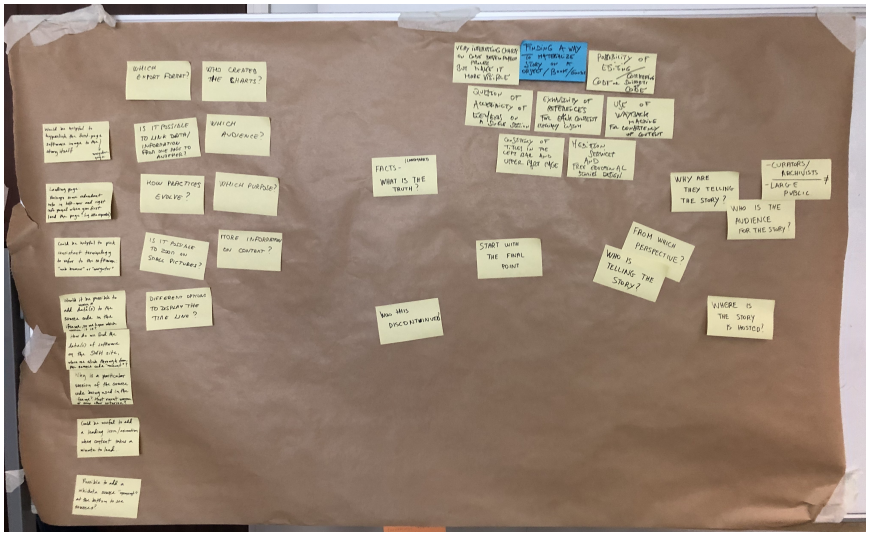

Figure 14: Feedback on Amaya story

Questions/feedback regarding the tool

- Would be helpful to hyperlink the first-page software image to the story itself
- Landing page: perhaps some redundant info in left-nav and right info panel when you first load the page (e.g. title repeated)
- Which export format?
- Is it possible to link data/information from one page to another?
- Is it possible to zoom in small pictures?
- Different options to display the timeline?
- Could be useful to add a leading icon/animation when content takes a minute to load.
- Question of accessibility of elements on a single section
- Finding a way to materialize story on object/book/goodie

- Possibility of editing/commenting code or snippet of code

Questions/feedback regarding the story itself

- More information on context?
- Facts - what is the truth?
- Was this discontinued?
- How practices evolve?
- Who created the charts?
- Which audience?
- Who is the audience of the story?
- Curators/archivists vs. large public
- Which purpose?
- Who is telling the story? From which perspective?
- Why are they telling the story?
- Where is the story hosted?
- Could be helpful to pick consistent terminology to refer to the software: "web browser" or "navigator"
- Would it be possible to add version numbers and dates to the source code in the iframe, so we know which version it is?
- How do we find the date(s) of software on the SWH site, when we click through from the source code "moment"?
- Why is a particular version of the source code being used in the iframe? Most recent version or some other criterion?
- Possible to add a Wikidata source "moment' at the bottom to see sources?
- Very interesting charts on code development, but have it more visible
- Consistency of titles in the left bar and upper right page
- Exhaustivity of references for each content, especially custom
- Use of Wayback machine for consistency of content

6.4 Synthetizing our thoughts on software story telling

As a last activity to help us synthetize our thoughts on presenting software we worked on four different questions:

- What are good angles to present a software?
- What to ask when interviewing a historical developer of the software?
- How to present a piece of code?
- Looking back on the guidebook, what materials would you include in collecting efforts?

We worked in small groups (in a short amount of remaining time). Resulting contributions can be found below, both as pictures and transcript of the result.

WHAT ARE GOOD ANGLES TO PRESENT A SOFTWARE?

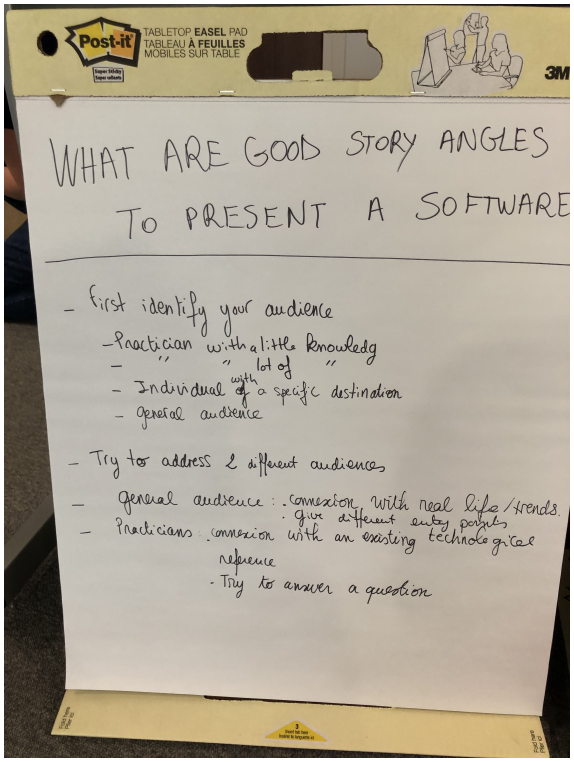

Figure 15: What are good angles to present a software?

- First identify your audience
  - Practician with a little knowledge
  - Practician with a lot of knowledge
  - Individual with a specific destination/interest
  - General audience
- Try to address two different audiences
- General audience
  - Connexion with real-life trends
  - Give entry points
- Practicians
  - Connexion with an existing technology
  - Try to answer a specific question

WHAT TO ASK WHEN INTERVIEWING A HISTORICAL DEVELOPER OF THE SOFTWARE?

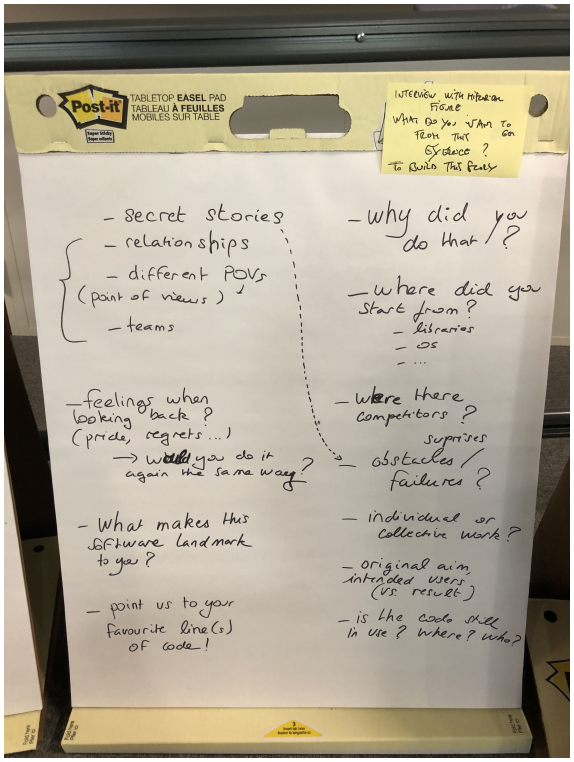

Figure 16: What to ask when interviewing a historical developer of the software?

- Why did you do that?
- Where did you start from? (libraries, OS..)
- Original aim, intended users (vs. result)
- Were there competitors
- Secret stories
- Surprises / obstacles /
- failures
- Relationships
- Different point of views
- Teams, Individual or collective work?
- Feelings when looking back

- Pride, regrets, would you do it again?
- What makes this software landmark for you?
- Point us to your favourite line of code!
- Is the code still in use? Where? Who?

HOW TO PRESENT A PIECE OF CODE?

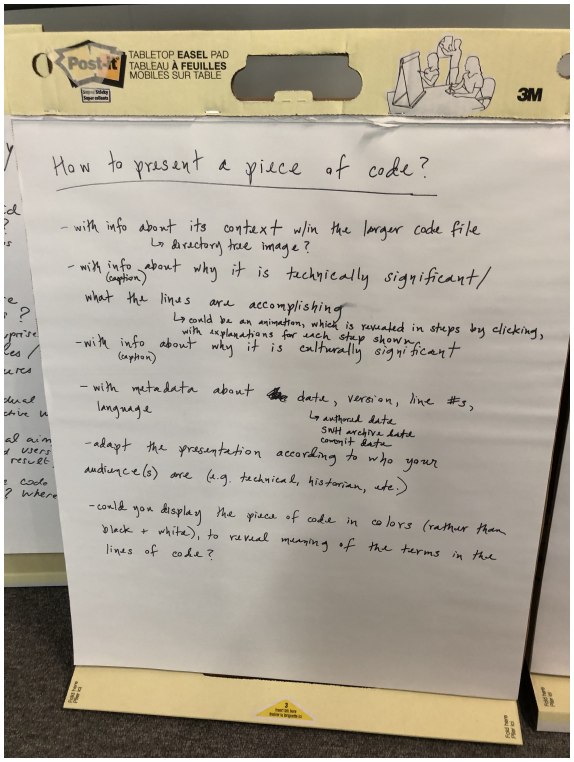

Figure 17: How to present a piece of code?

- With info about its context within the larger code file. (directory tree image?)
- With info regarding why it is technically significant / what the lines are accomplishing
- Could be an animation, which is revealed in steps by clicking, with explanations for each steps shown
- With info regarding why it is culturally significant
- With metadata about dates (authored date, SWH archive date, commit date), version, line numbers
- Adapt the presentation according to who your audience(s) are (e.g. technical, historians etc.)
- Display the piece of code in colours to reveal meaning of the terms in the line of code.

LOOKING BACK ON THE GUIDEBOOK, WHAT MATERIALS WOULD YOU INCLUDE IN COLLECTING EFFORTS?

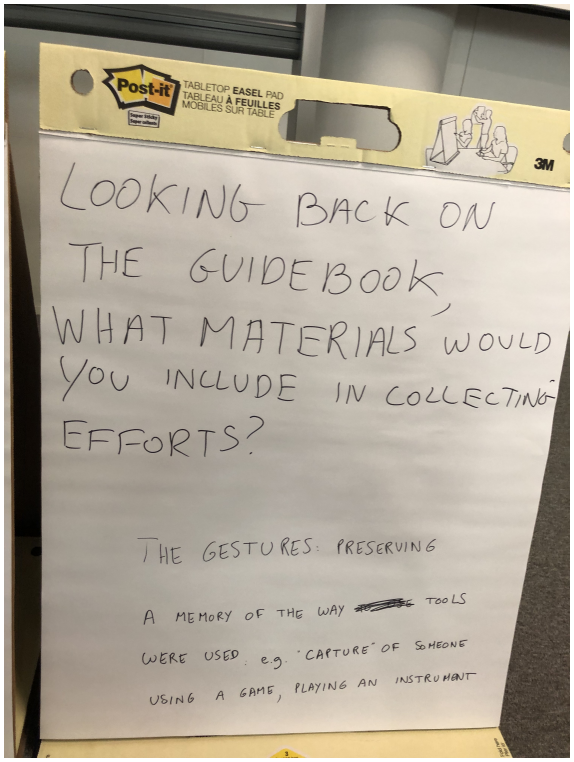

Figure 18: Looking back on the guidebook, what materials in collecting efforts?

- The gestures: preserving the way tools were used
- E.g capture of someone using a game, playing an instrument

7 Conclusion and next steps

In these two days of intense work and exciting discussions we touched upon a diversity of topics regarding software preservation and presentation, from very technical ones to more philosophical ones. Many questions were raised, some answers were sketched. One thing is certain: legacy software conservation is still an emerging issue, and many challenges still lie ahead of our community. As a wrap-up activity, we each identified topics we would like others to work in in the next six months (on red post-its) and things we would like ourselves to do in the next six months (on green post-its). The result can be seen in the picture and transcript below. Note that the answers were anonymous.

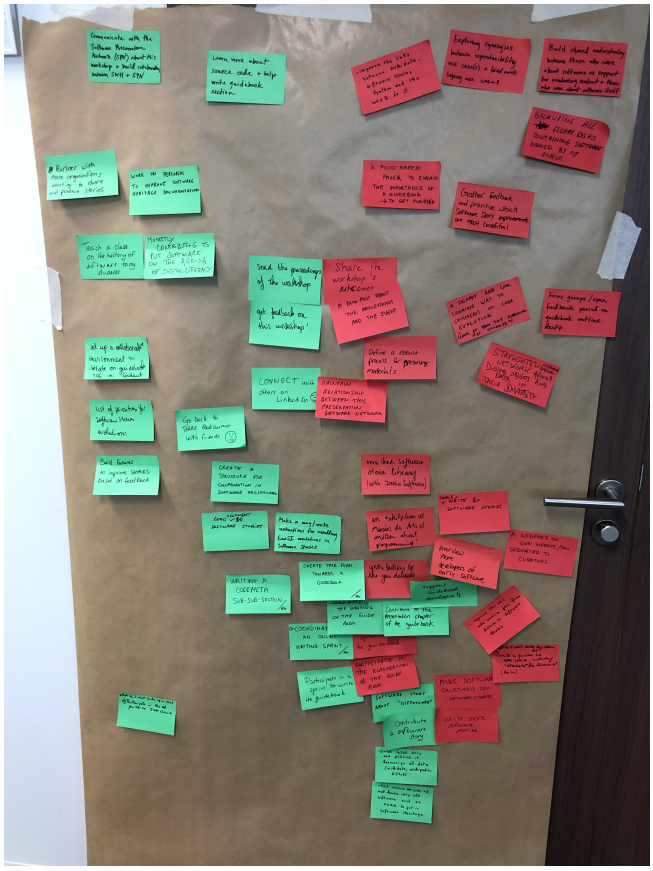

Figure 19: Result of the wrap-up session

Things I would like see happend in the next six months

- Improve the links between Wikidata, Software Stories system and the way to do it
- Explore synergies between reproducibility use case(s) + landmark legacy use cases
- Build common understanding between those who care about software as support for rendering content + those who care about software itself
- Backuping all the floppy disks containing software owned by my museum
- A policy makers paper to explain the importance of a guidebook, to get funding
- Gather feedback and priorities, which Software Story improvement are most beneficial
- Share the workshop's outcome
- A blog post about the proceedings and the event
- A snappy and cool looking way to comment on code evolution ("look at how this function has changed")
- Focus groups / open feedback period on guidebook outline draft
- Strengthen network around digital objects and data in their diversity
- Define a robust process for preserving materials
- Maintain relationship between this preservation software network
- Enrich software stories library (with Inria software)
- Choose and write 10 software stories
- An exhibition at Musée des Arts et Métiers about programming!
- Contributions to the guidebook
- Interview more developers of early software
- Improve the way source code goes from Github to software stories
- Write a guide to Software Stories, including "Wikidata for dummies"
- Organize a sprint to write the guidebook
- Participate in the elaboration of the guidebook
- More software collections on Software Stories
- Write more Software Stories

Things I would like to do in the next six months

- Communicate with the software preservation network (SPN) about this workshop and build collaboration between SWH and SPN
- Partner with more organizations wanting to produce and share stories
- Work on personas to improve Software Heritage documentation
- Teach a class on history of software to my students
- Modestly contribute to put software on the agenda of digital literacy
- Send the proceedings of the workshop
- Get feedback on this workshop!
- Set up collaborative environment to iterate on guide TOC and content
- List of priorities for software stories evolutions
- Go back to Terre restaurant with friends ;)
- Connect with others on LinkedIn :)
- Build features to improve Stories based on feedback
- Create a structure for collaboration in software preservation
- Read and comment 10 Software Stories
- Make a way / write instructions for embedding EaaSI emulations in Software Stories
- Writing a CodeMeta sub-sub-section
- Create the plan towards a guidebook
- Co-coordinating an online writing sprint
- Coordinating the writing of the guidebook
- Contributing to the presentation chapter of the guidebook
- Support guidebook developments
- Participate in the guide to Software Stories
- Write a software story about "Diffoscope"
- Contribute a software story
- Finish Scilab story and publish it, harmonize all data
- Collect source code of well known very old software such as ELIZA and post in Software Heritage

Appendix 1 - Guidebook chapters pictures

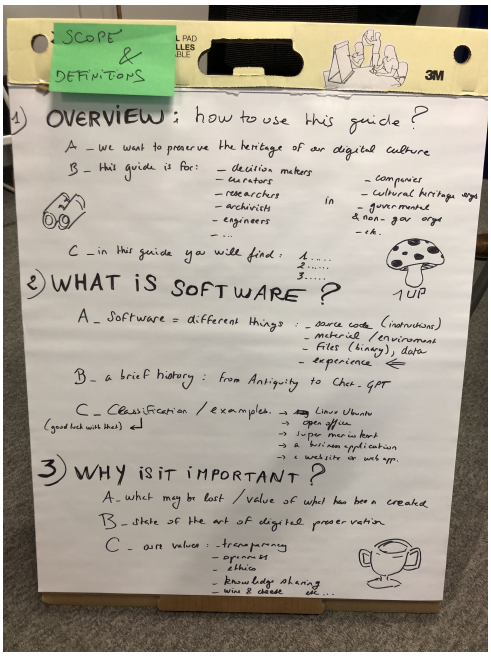

Figure 20: Chapter 1 - Intro

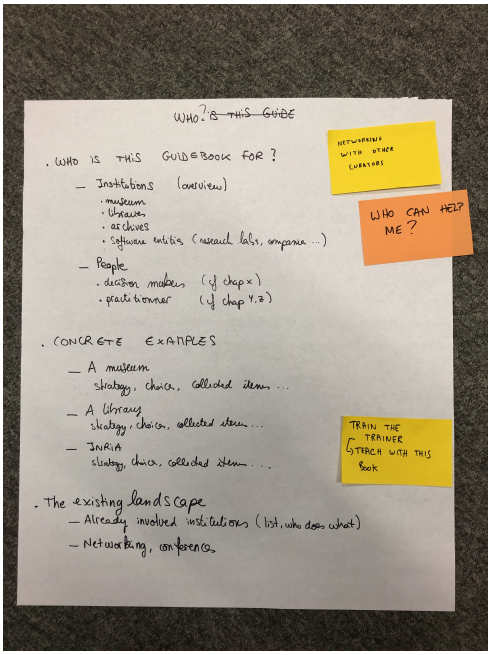

Figure 21: Chapter 2 - Who? Uses Cases

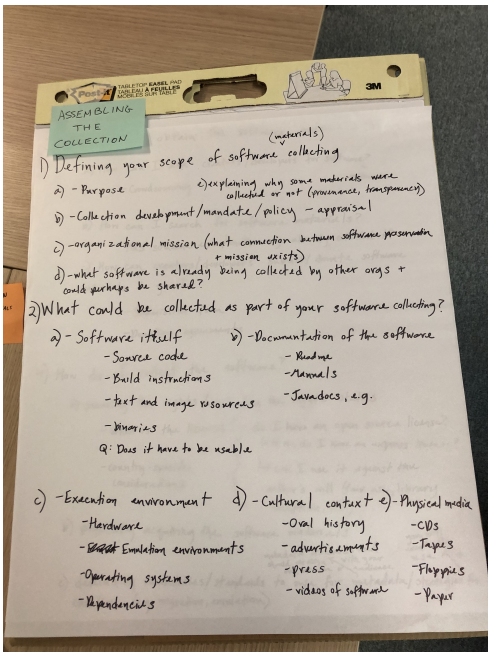

Figure 22: Chapter 3 - Assembling the collection (1/2)

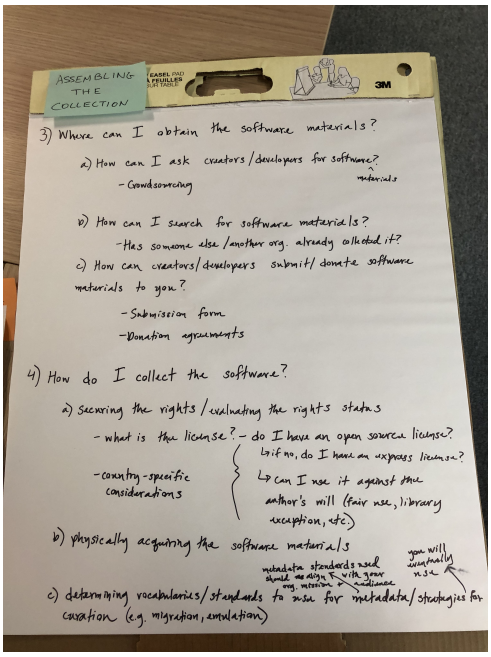

Figure 23: Chapter 3 - Assembling the collection (2/2)

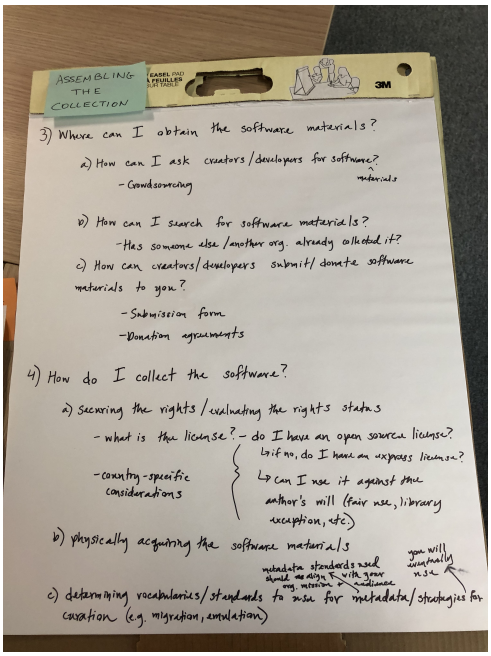

Figure 24: Chapter 4 - Processes (1/3)

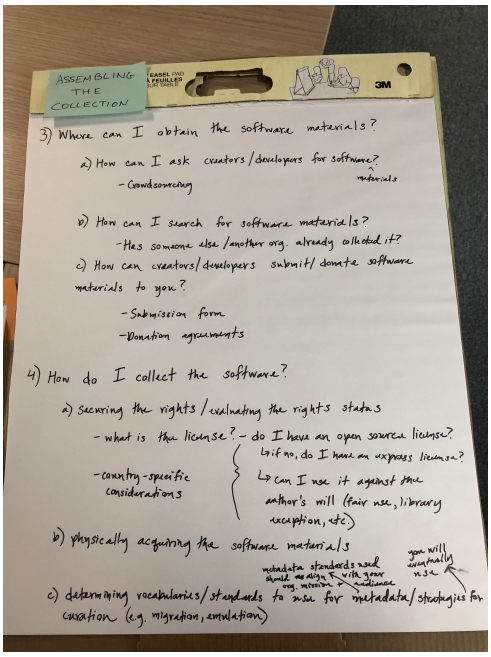

Figure 25: Chapter 4 - Processes (2/3)

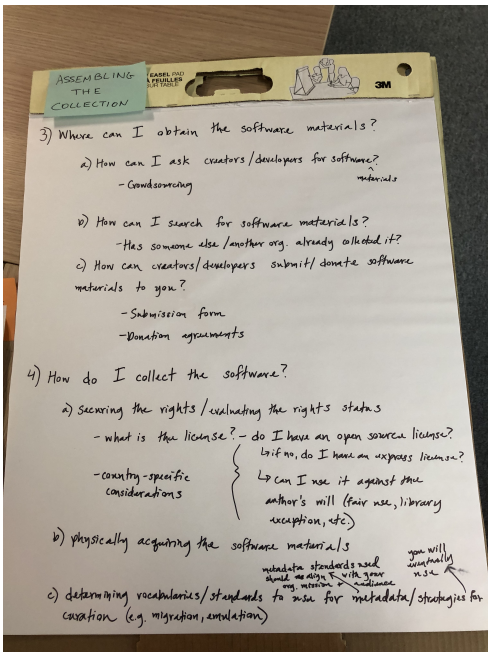

Figure 26: Chapter 4 - Processes (3/3)

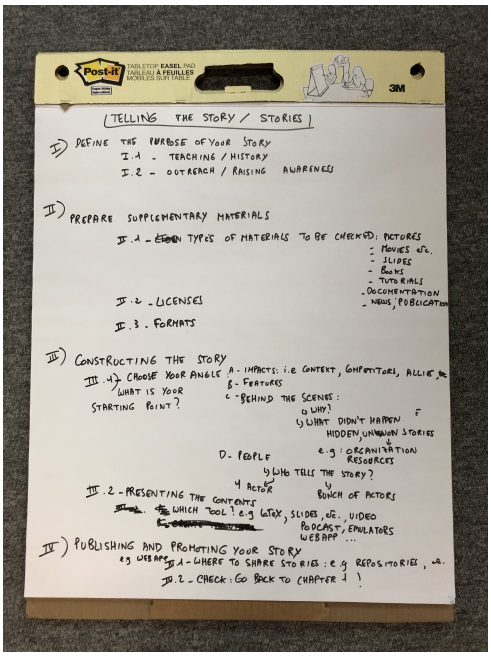

Figure 27: Chapter 5 - Telling the story/stories

Appendix 2 - Guidebook detailed table of content

LANDMARK LEGACY SOFTWARE PRESERVATION GUIDEBOOK

1. **Chapter 1 - Scope and Definition**
    a) Overview: how to use this guide
        i. Why do we want to preserve the heritage of our digital culture?
        ii. Who is this guide for?
            - Decision makers
            - Curators
            - Researchers
            - Archivists
            - Engineers
            - ...
            - In Companies
            - Cultural heritage organizations
            - Governmental and Non-governmental organizations
        iii. What you will find in this guide
    b) What is software?
        i. Software = different things
            - Source code (instructions)
            - Material / environment
            - Files (binaries), data
            - Experience
        ii. A brief story: from Antiquity to Chat-GPT
        iii. Classification (good luck with that) / examples
            - Linux Ubuntu
            - Open office
            - Supermario kart
            - A business application
            - A website or web app
    c) Why is it important?
        i. What may be lost / value of what as created

ii. State of the art of digital preservation
iii. Our values
- Transparency
- Openness
- Ethics
- Knowledge sharing
- Wine and cheese etc.

2. **Chapter 2 - Use cases and existing landscape**
   a) Concrete examples
      i. A museum
      ii. A library
      iii. A software producer (ex: Inria)
      iv. An individual owning legacy software
   b) The existing landscape
      i. Already involved institution. List, contact point, who does what
      ii. Networking, conferences

3. **Chapter 3 - Assembling the collection**
   a) Defining your scope of software collecting
      i. Purpose
      ii. Collection development / mandate / policy - appraisal
      iii. Organizational mission: what connection between software preservation and organization mission exists
      iv. What software is already being collected by other orgs and could perhaps be shared
   b) What could be collected as part of your software collecting?
      i. Software itself
         - Source code
         - Build instructions
         - Text and image ressources
         - Binaries (Q: does it have to be usable?)
      ii. Documentation of the software
         - Readme

- Manuals
- Javadocs, e.g.

iii. Execution environment
- Hardware
- Emulation environments
- Operating systems
- Dependencies

iv. Cultural context
- Oral history
- Advertisements
- Press
- Videos of software

v. Physical media
- CDs
- Tapes
- Floppies
- Paper

c) Where can I obtain the software materials

i. How can I ask creators/developers for software materials?
- Crowdsourcing,...

ii. How can I search for software materials?
- Has someone else/another org. already collected it?

iii. How can creators/developers submit/donate software materials to you?
- Submission form
- Donation agreements

d) How do I collect the software?

i. Securing the rights / evaluating the rights status
- What is a license? Do I have an open source license? If no, do I have an express license? Can I use it against the author's will (faire use, library exceptions etc.)
- Country specific considerations

ii. Physically acquiring the software materials

iii. Determining vocabularies/standards to use for metadata / strategies for curation (e.g. migration, emulation)
    - Metadata standards used should align with your org. mission + audience

4. **Chapter 4 - The process(es)**
    a) Curating software source code
        i. Process overview
        ii. The phases
        iii. The roles and skills
        iv. The tools
        v. A complete example of the process
    b) Curating the executables
        i. Environments
        ii. Emulated software
    c) Curating physical software artifacts
5. **Chapter 5 - Telling the story/stories**
    a) Define the purpose of your story
        i. Teaching / history
        ii. Outreach / raising awarenesss
    b) Prepare supplementary materials
        i. Types of materials to be checked
            - Pictures
            - Movies
            - Slides
            - Books
            - Tutorials
            - Documentation
            - News, Publications....
        ii. Licenses
        iii. Formats
    c) Constructing the story
        i. What is your angle? Starting point?

- Impacts: i.e context, competitors, allies
- Features
- Behind the scenes: Why? What didn't happen, hidden, unknown stories
- People: who tells the story? One actor vs. a bunch of actors

ii. Presenting the contents

- Which tools? E.g. Latex, slides, video, podcast, emulators, webapp

d) Publishing and promoting your story

i. Where to share stories

ii. Check: go back to chapter one!

6. **Bibliography**

References

[1] "Swhap days 2022 webpage." available at https://www.softwareheritage.org/news/events/swhap_days_2022/.

[2] M.Gruenpeter, K. Thornton, K. Seals-Nutt, C. Montangero, and G. Scatena, "Software stories for landmark legacy code," 2023.

[3] L. Bussi, R. D. Cosmo, C. Montangero, and G. Scatena, "The software heritage acquisition process," Tech. Rep. CI-2019/WS/8, UNESCO, Università di Pisa, Inria, 2019.

[4] Expert Group Report, "Paris call: Software source code as heritage for sustainable development," 2019. Available at https://unesdoc.unesco.org/ark:/48223/pf0000366715.

[5] "Software stories website." available at ttps://stories.softwareheritage.org/.

[6] "Preserving our software heritage." available at https://annex.softwareheritage.org/public/talks/2023/2023-09-14-ShwapStoriesWksp.pdf.

[7] R. D. Cosmo and S. Zacchiroli, "Software heritage: Why and how to preserve software source code," in *iPRES 2017: 14th International Conference on Digital Preservation*, (Kyoto, Japan), 2017.

[8] "Science stories website." available at https://sciencestories.io.

[9] T. Adlin and J. Pruitt, *The essential persona lifecycle: Your guide to building and using personas*. Morgan Kaufmann, 2010.

[10] "Introducing the Smalltalk Zoo," Dec. 2020. Available at https://computerhistory.org/blog/introducing-the-smalltalk-zoo-48-years-of-smalltalk-history-at-chm/.

[11] "*Pac-Man*," Oct. 2023. available at https://en.wikipedia.org/w/index.php?title=Pac-Man&oldid=1180092823.

[12] "ELIZA," Oct. 2023. available at https://en.wikipedia.org/w/index.php?title=ELIZA&oldid=1179533368.

[13] "Introducing ChatGPT." available at https://openai.com/blog/chatgpt.

[14] "History of the World Wide Web," Oct. 2023. available at https://en.wikipedia.org/w/index.php?title=History_of_the_World_Wide_Web&oldid=1179905123.

[15] “Mosaic (web browser),” Sept. 2023. available at https://en.wikipedia.org/w/index.php?title=Mosaic_(web_browser)&oldid=1177281756.

[16] “Fortran,” Oct. 2023. available at https://en.wikipedia.org/w/index.php?title=Fortran&oldid=1179631697.

[17] “Wikidata for digital preservation portal.” available at https://wikidp.org.

[18] K. Thornton and K. Seals-Nutt, “Science stories: Using iiif and wikidata to create a linked-data application.,” in *International Semantic Web Conference (P&D/Industry/BlueSky)*, 2018.

[19] “Grace hopper science story.” available at https://sciencestories.io/Q11641.

[20] “django-wikidata-api: Python Package for interfacing with Wikidata within a Django App.” SWHID:<swh:1:dir:989ae3116da9f3f206e18e2c6f02289bfc25b544> available at https://archive.softwareheritage.org/swh:1:dir:989ae3116da9f3f206e18e2c6f02289bfc25b544;origin=https://github.com/kennethsn/django-wikidata-api;visit=swh:1:snp:8db8fc7e1b25de3e6aed11f2bed313d8160573ea;anchor=swh:1:rev:5efb575cf9876d42d34ac4b5b286d39f24470ade.

[21] K. Seals-Nutt, “react-story-api: React components for science stories,” Nov. 2022. SWHID:<swh:1:dir:b1ea18251194d8eb74d5308e92a24d23bb7c35b9> available at https://archive.softwareheritage.org/swh:1:dir:b1ea18251194d8eb74d5308e92a24d23bb7c35b9;origin=https://github.com/kennethsn/react-stories-api;visit=swh:1:snp:d056ceaf25ea31b72536fca407ff58e905df3320;anchor=swh:1:rev:5a9101ab2ad7c7c3bf685706e91df890d28f0f15.

[22] “Amaya software story,” Oct. 2023. available at https://stories.softwareheritage.org/inria/Q455973.