United Nations
Educational, Scientific and
Cultural Organization

Memory of
the World

inventors for the digital world

PARIS CALL
SOFTWARE SOURCE CODE
AS HERITAGE FOR SUSTAINABLE DEVELOPMENT

Software Heritage

The designations employed and the presentation of material throughout this publication do not imply the expression of any opinion whatsoever on the part of UNESCO concerning the legal status of any country, territory, city or area or of its authorities, or concerning the delimitation of its frontiers or boundaries.

The ideas and opinions expressed in this publication are those of the authors; they are not necessarily those of UNESCO and do not commit the Organization.

License: IGO CC-BY 4.0

CLD 2727_18 CI-2019/WS/3

Table of Contents

FOREWORD

An eminent group of international experts met upon invitation of UNESCO and Inria on 6 and 7 November 2018 at UNESCO's Headquarters to discuss software preservation. Indeed, it is increasingly becoming a significant means for capitalizing on the knowledge built over humankind's recent history in order to foster innovation and advance our understanding of ourselves and of our environment.

In a world where digital technology has become for many an essential tool for social existence, communication, creation, sharing, and is increasingly indispensable for accessing public services, the role of software development is still largely underrated, as is the recognition of software source code as an intellectual effort and as the receptacle and expression of part of our knowledge.

That is why it is crucial to work towards preserving the technological and scientific knowledge embodied in software source code. This objective is at the core of the cooperation between UNESCO and Inria, which started in 2017, in line with UNESCO's concept of inclusive knowledge societies based on four pillars: freedom of expression, universal access to information and knowledge, respect for cultural and linguistic diversity, and quality education for all.

The expert meeting provided a unique platform to engage with the eminent experts, practitioners, policymakers and activists representing different disciplinary and professional worlds. This includes communities engaged in the preservation, archiving, and dissemination of documentary heritage, particularly in the context of UNESCO's Memory of the World Programme; the technical community, ranging from research to computer science education; and activists calling for an open and free environment for cultural expression and technological innovation.

Emerging from this meeting is the Paris Call, representing an important first step towards a coordinated response to the challenges that have been identified. These challenges include the importance of raising awareness among decision-makers, and the recognition of software creators as well as of the contribution of women and minorities to digital innovation and software. The Call goes further to argue for greater access to software source code in order to ensure that citizens, and young people in particular, are empowered with sufficient digital skills and literacy to fully participate in today's digital society.

Accordingly, the annexed report highlights the importance of preserving software in general, and software source code in particular, as important levers for sustainable development.

These efforts, however, are just starting. It is our collective responsibility and we all must ensure that the knowledge accumulated – and constantly being generated – is not lost. This focus on preservation enables us all to embrace software source code as part of the heritage of our knowledge societies. The Paris Call thus becomes a strong basis to imagine and build new actions around the preservation of the code, and sustain Free and Open Source Software communities through the exchange of the knowledge now archived as part of the Software Heritage Initiative.

Moez Chakchouk
Assistant Director-General
for Communication and Information
UNESCO

PARIS CALL

"Software Source Code as Heritage for Sustainable Development"

Part of our Heritage, Pillar of our Present, Enabler of our Future

Considering the results of our consultations as reflected in the Annexed report,
We, the participants at the UNESCO/Inria Expert Meeting, held in Paris, France, 6-7 November, 2018,

1. Considering software source code as a key component of human creativity, sustainable development, society and culture;
2. Recalling the 2003 Charter on the Preservation of Digital Heritage;
3. Recalling further the 2011 Moscow Declaration on Digital Information Preservation;
4. Recalling also the 2012 UNESCO/University of British Columbia (UBC) Vancouver Declaration (on Memory of the World in the context of digitization and preservation);
5. Recalling also the 2015 Recommendation Concerning the Preservation of, and Access to, Documentary Heritage, Including in Digital Form, that recognises the value of open source software and open standards for long term preservation;
6. Emphasising the importance of software source code for a transparent society;
7. Emphasising also that software source code is an essential pillar of education and research;
8. Emphasising also the centrality of software to modern commerce and industry especially as a medium for innovation and economic development;
9. Recognising the growing importance of free and open source software, with humankind constantly creating an unprecedented software commons;
10. Recognising also that the preservation and sharing of software source code is threatened by a lack of awareness of its nature and role as well as a lack of preservation infrastructure;
11. Welcoming the United Nations (UN) 2030 Agenda for Sustainable Development[1], particularly its focus on strengthening "efforts to protect and safeguard the world's cultural [...] heritage" as well as on ensuring public access to information and protecting fundamental freedoms;
12. Welcoming also the Memorandum of Understanding between UNESCO and Inria as an important lever in supporting the identification, preservation and promotion of software source code as digital heritage for Sustainable Development.

We therefore:

→ **Call on each UNESCO Member State to:**

13. **Recognise** software source code as a precious asset of humankind, intersecting with human creativity, development, society and culture;
14. **Recognise** software source code as a fundamental enabler in all aspects of human endeavour;
15. **Recognise** software source code as a fundamental research document on a par with scholarly articles and research data;
16. **Recognise** that the source code of software used for the implementation of laws and regulations defines the experience of the law by citizens;
17. Create an enabling legal, policy and institutional environment where software source code can flourish as an integral part of knowledge societies;
18. **Integrate** the scientific fundamentals of computing/informatics within general education for all citizens;
19. **Support** the development of shared infrastructures to collect, preserve and make available software source code;
20. **Establish** an open and international research infrastructural framework for the large scale analysis and improvement of the quality, safety and security of the software commons;

21. **Ensure** necessary exceptions to copyright and limitations on intermediary liability related to software for archival, preservation, accessibility, education and research purposes;
22. **Enable** effective independent auditing of software source code used to make decisions that may affect fundamental rights of human beings and where possible ensure it is made available under an open source license;
23. **Implement**, with support from UNESCO's Memory of the World Programme, the 2015 Recommendation concerning the Preservation of, and Access to, Documentary Heritage, including in Digital Form, inviting inter-alia Member States to facilitate access to proprietary codes, keys and unlocked versions of technology on a nonprofit basis.

→ **Call on UNESCO and Inria to:**

24. **Strengthen** UNESCO's support for the Software Heritage initiative, as a way of enhancing awareness of the importance of preserving and providing access to source code;
25. **Forge** more strategic partnerships in order to create greater recognition of software development activity as science and research, particularly by demonstrating how software source code can be appropriated as a research product worthy of preservation, while at the same time promoting its recognition as a valid field of both applied and academic enquiry, with reproducible or verifiable research results;
26. **Support** efforts for the development of an open Global Software Registry, which will help all stakeholders to recognize and enable software reuse as an important part of all modern software developments by providing a universal catalogue that will index all available software components, with the metadata needed to properly locate and reuse them.

→ **Call on software developers, memory institutions, the business sector, academia and civil society, within their competency, to:**

27. **Recognise** that software is the result of a significant part of the intellectual efforts of humankind over recent decades, and it is an important part of our cultural and industrial heritage;
28. **Support** efforts to gather and preserve the artifacts and narratives of the history of computing, while the earlier creators are still alive;
29. **Promote** software development as a valuable research activity, and research software as a key enabler for Open Science/Open Research, sharing good practices and recognising in the careers of academics their contributions to high quality software development, in all their forms;
30. **Recognize** the importance of contributions by people of all genders from all over the world to the software commons, supporting a diverse and inclusive environment for all aspects of software development and curation;
31. **Educating** decision makers on the specificities of software, and software source code in particular, raising awareness about the threats to the software commons and the importance to protect it;
32. **Encourage** all stakeholders to develop a common system of cataloguing to allow for easy identification and retrieval of software source code, even across the many platforms and infrastructures used to develop and/or distribute it;
33. **Support** stakeholders in developing a universal archive, as part of a broad effort at digital preservation, that will ensure persistence of and universal access to software source code;
34. **Encourage** multidisciplinary activity in the field of software preservation, and in particular collaboration with the humanities and social sciences whose contributions are essential to study the history of technology;
35. **Adapt** processes, workflows and licensing schemas in the software industry to ease the transition of future proprietary software source code into the software commons once it is no longer commercially viable;
36. **Foster** international collaboration to build a common framework for software preservation and access, and mutualise resources, in order to avoid the dispersion of efforts;
37. **Promote** the recording of the activity of software developers, captured as documentary heritage either in analogue or digital form, which are suitable for preservation in their own right and ensure that they are linked with the source code.
38. **Support** all stakeholders in developing the understanding that software source code is intertwined more and more with the fabric of our society, hence the utmost care needs to be used during its development process to manage its potential consequences on society and people.

Adopted on 7 November 2018, Paris, France

SIGNATORIES

Abramatic, Jean François	Inria	Emeritus Senior Scientist
Albert, Kendra	Software Preservation Network	Legal Advisor
Alberts, Gerard	University of Amsterdam	Historian
Bradley, Kevin	Australian Collection and Reader Services, National Library of Australia	Assistant Director-General
Buckley, Robert	PERSIST Policy Working Group	Chair
Chue Hong, Neil	Software Sustainability Institute, University of Edinburgh	Director
Clipsham, David	The National Archives (UK)	Digital Preservation and Archiving Technical Architect
Cochrane, Euan	Yale University	Digital Preservation Manager
Di Cosmo, Roberto	Software Heritage, Inria	Director
Duplouy, Laurent	Multimedia collection, BNF	Curator
Gharsallah, Mehdi	French Ministry of Higher Education, Research and Innovation	Conseiller stratégique pour le numérique
Greenberg, Joshua M.	Alfred P. Sloan Foundation	Program Director, Digital Information Technology
Guerry, Bastien	Free Software at Etalab (France)	Head
Hinchey, Mike	International Federation for Information Processing	President
Inverardi, Paola	University of L'Aquila (Italy)	Rector
Issarny, Valerie	ACM-Europe and Inria	Computer scientist

Madhavan Pillai, Arun	ICFOSS, Government of Kerala, India	Program Head
Marzano, Flavia	Roma Semplice	Councilor
Miura, Grégory	Bordeaux Montaigne University	Director of the Shared Documentation Service
Moreau, Patrick	The French National Centre for Scientific Research	Industrial Partnership Manager
Nardelli, Enrico	Informatics Europe	President
O'Donohue, Pearse	Future Networks Team in DG CONNECT at the European Commission	Director
Osuna Alarcón, Maria R.	University of Salamanca	Senior Lecturer
Palm, Jonas	SCOT, Memory of the World Programme	Chair
Perelmuter, Tanya	Software Heritage Foundation	Strategic Partnership Director
Phipps, Simon	Open Source Initiative	President
Piana, Carlo	Array	Lawyer
Rugier, Nicolas	Inria	Researcher
Sassi, Melissa	IEEE Digital Skills Working Group	Chair
Schüller, Dietrich	Austrian National Committee	Chair
Seles, Anthea	International Council on Archives (ICA)	Secretary-General
Shustek, Len	Computer History Museum	Chairman
Smith, Arfon	Space Telescope Science Institute	Head of the data science mission office
Webb, Mary	International Federation for Information Processing	TC3 member
Wheatley, Paul	Digital Preservation Coalition	Head of Research and Practice
Wyber, Stephen	IFLA	IEEE Digital Skills

REPORT FROM THE EXPERT GROUP ON SOFTWARE SOURCE CODE AS HERITAGE FOR SUSTAINABLE DEVELOPMENT

UNESCO Headquarters, 6-7 November 2018

1. Introduction

Software embodies a vast part of our knowledge and cultural heritage. It mediates access to all digital information, supports and embodies new forms of social and political organisations, powers our industries and innovation, and is a pillar of research. It binds together the personal, social, industrial, and digital aspects[1] of our lives.

Software is an essential mediator to access all digital information and hence a key component to fulfill the human right of *access to information*.[2]

Software embodies the procedures by which the citizen engages with the state, through which the citizen and the market interact and in which citizens communicate with each other and enjoy cultural and leisure pursuits. Our ability to see society in action and guarantee the democracy that sustains it is increasingly dependent on our ability to review the software by which it is enabled at every level.

Software both runs factories and is itself a creative industry. Industry increasingly and sometimes completely relies upon custom built software for logistics, factory production, financial and human resources management and education. The advent of software has led to increased ability to innovate. At the same time, cultural products, such as video games, artistic works and modern cinema are nowadays created with sophisticated software tools. These works are then preserved for later access by libraries, archives and museums using software systems.

Software is a fundamental pillar of modern research[3], across all fields and disciplines. Ensuring transfer of knowledge and understanding of software programs is critical for both computing experts and for users of computing techniques in other areas of research, education, and applications. That adds to the concerns of the Memory of the World Programme to protect our digital heritage from obsolescence and make it accessible to the future generations.

Looking more closely, though, the actual knowledge embedded in software is not just contained in executable binaries, which are designed to run on specific hardware and software platforms and are almost incomprehensible to human beings[4]. Rather, knowledge is contained also in software source code which is, as eloquently stated in the very well crafted definition found in the GNU General Public License,[5] "the preferred form [of a program] for making modifications to it [as a developer]".

Software source code is the result of a significant part of the intellectual creative efforts of humankind over the last decades, and is becoming an important part of our cultural heritage. Our ability to use, understand, adapt, correct, and evolve the devices on which our lives have come to depend relies on our ability to access, understand, adapt, correct, and evolve the software that controls them.

Thus, preserving software source code and making it widely available is vital to human cultural heritage and key to sustainable development.

2. Software source code: a precious asset of humankind

Software source code is a unique form of knowledge which is designed to be understood by a human being, the developer and, at the same time, easily converted into machine executable form.

2.1 A key component of human creativity, development, society and culture

Authoritative voices have spoken eloquently of the importance of source code. As far back as the 1980's Harold Abelson wrote "Programs must be written for humans to read",[6] and one of the founding fathers of computer science, Donald Knuth, wrote in one of his essays "Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do"[7].

As a digital object, software source code is subject to a specific workflow: it is routinely updated to cater to new needs and evolving contexts. In order to understand software source code, it is essential to have access to its entire development history. Thus, quite differently from other forms of knowledge, we have grown accustomed to using version control systems in order to track source code development and provide precious insight about its evolution. As Len Shustek puts it, "Source code provides a view into the mind of the designer".[8]

Software source code is now a relevant part of our information commons—the software commons,[9] i.e., the body of software that is widely available and can be reused and modified with minimal restrictions[10]. The rise of Free/Open Source Software[11] (FOSS) over the past decades has contributed enormously to this commons, opening up unprecedented collaboration opportunities, and fostering source code accessibility though its founding principles.

2.2 A needed prerequisite in many fields of activity

Software source code is not only an important part of our cultural heritage. Having access to the source code of a running piece of software is also a necessary prerequisite in many fields of human activity, in particular:

→ Software quality for society

Software quality, safety and security are of paramount importance today. Software is increasingly a central part of complex systems that lie at the heart of our transportation, energy, media, communication and financial infrastructure, as well as health care systems and personal medical devices, some of which may be embedded in our bodies, like pacemakers. Ensuring quality, safety and security of these systems relies on our ability to ensure quality, safety and security of all the software components they are built upon.[12] Since many modern systems are created by extensively reusing pre-existing building blocks, often available as FOSS, access to the source code of these building blocks is essential for the shared efforts to improve their quality, and hence the quality of the software as a whole.

→ Accountability of public administration and powers

Public administrations use software to collect and process public information about the domain they administer and personal information about their citizens. Along the lines of the principles set forth in the Open Government Partnership[13], public administrations need to make publicly and durably available the source code of the software they use, in order to be held accountable of their operations.

→ Transparency of automated decision making

Many decisions that have a direct impact on human beings and society are now taken automatically by computer systems[14], raising deep concerns and high expectations about the accountability and transparency of these decisions[15,16]. Ready access to the source code of the software involved in automated processes is one of the prerequisites to ensure transparency and accountability.[17]

→ **Reproducibility of research**

Scientific results are essential for the progress and evolution of humankind. They increase our knowledge, enable industrial evolution, and inform public policies. Most modern scientific results rely on software tools[18]. Availability of the source code of these software tools is one of the prerequisites to ensure that results can be reproduced[19], understood and trusted. Ensuring transparency of the scientific process is essential as it eases the acceptance of research results.

2.3 An essential pillar of education and research

We acknowledge the existing effort to identify and address the digital divide, in particular between those having access to ICT and those that lack it[20]. Here we go one step forward, and remark that even among those that have full access to ICT, a second divide is emerging, impacting the potential abilities of the citizens to fully participate in the digital society. This is a divide between 'digitally savvy' who know not only how to use, but also how to develop and adapt software tools, and those who are primarily consumers of software technology developed by others. This widening gap is especially concerning, given the increasing complexity of the technology that is broadly deployed, like artificial intelligence (AI) systems.

In order to reduce this divide, a general formal and informal education in the fundamentals of computer science/informatics, including computer programming, is needed[21,22]. In the years to come, basic literacies like reading, writing and numeracy, that everyone should master in childhood, will be joined by the fundamentals of computer science and informatics. To support this effort, several aspects need to be taken into account.

Physical libraries are essential to help people learn to read, but also support advanced studies of literature; by analogy, curated collections of source code can be used to teach basic programming skills and can also inspire deeper learning of the fundamentals of computer science.

- → Students should be provided with authentic experiences in software development as well as an understanding of the design and implementation process, including its underpinnings in algorithms and logic.

- → It can be valuable to record narratives and discussions among designers, who played a fundamental role in the creation of modern technology.

- → Educating all citizens, but especially computer science and informatics students about how to make ethical choices about technological use.

For all these reasons, software deserves special care and attention. We need to ensure that the importance of software is recognized and that the software source code is properly preserved and made available to the future generations. We must build the necessary competencies, infrastructures and processes to sustain it and improve it over time[23].

3. Threats to software source code preservation and sharing

Software source code is a precious asset whose study and preservation is currently endangered: there is insufficient understanding of fundamental issues like the nature of software, and a lack of coherent technical and economic strategies, as well as significant barriers, both legal and logistical, to preserving and sustaining the knowledge embedded in software. Managing these threats requires a concerted effort. In some instances a societal change is needed, in others appropriate infrastructures will have to be built.

3.1 Insufficient awareness among decision makers

The first and the foremost threat to the software commons is a lack of understanding of the nature and inner workings of software and software source code by decision makers.

Generally, neither decision makers nor the general public are aware that software artifacts are radically different from any previous man-made creations. While these have been just amplifications of physical and sensory capabilities of people, software source code is a new form and representation of human knowledge. It is «actionable knowledge», that is ready to be executed on the appropriate hardware and that can dynamically interact with the world. Software source code is thus an unprecedented mechanization of human knowledge. Without a thorough comprehension of this concept the role of software in society cannot be really understood.

For example, the recent copyright reform in Europe, despite being explicitly intended to regulate multimedia and publishing industries, has been articulated in a way that is likely to impact the software development ecosystem presenting challenges to its very existence[24]. The debate that ensued has made it evident that few policymakers, or even their advisors, understand the nature of software, and that there is a need for a consistent and compelling voice to advocate for the interests of the software commons in the political arena.

At the same time, some decision makers have shown that they *understand the importance* of computing generally and source code specifically. In the United Kingdom, a computing programme of study has been incorporated in the national curriculum[25]. In the United States, both President Obama and President Trump have proposed more funding for computer science education.[26] UNESCO supports the Software Heritage initiative. Much more is to be done.

3.2 Specific legal challenges

Insufficient awareness among decision makers often produces accidental hurdles to the preservation of software source code and access to code. But beyond these unintentional challenges, many countries have legal rules that obstruct the passage of software into preserved heritage. Such regimes have arisen both from the implementation of treaty- and WTO-originated obligations and from the demands of specific commercial sectors feeling threatened by the emerging digital society.

Anti-circumvention provisions in many countries make it difficult to extract software source code without legal risk, and narrow specific copyright exemptions for preservation that ignore software can make it difficult to archive source code in countries without flexible fair use/ fair dealing doctrines. Such lack of legal provisions can dissuade risk-averse preservation institutions from even attempting to save source code.[27]

Special text and data mining rights may prevent or hamper large scale automated analysis of, and machine learning on, the source code corpus, which are already being used by practitioners and have shown that they are necessary to improve software quality, for the benefit of all of society.

3.3 Lack of recognition for software creators

Along with the insufficient understanding of software comes a widespread lack of recognition of the software development activity in general.

Software development is a human activity, involving a broad range of creators, ranging from software developers to system architects, from engineers to scientists. The authors of software systems deserve credit for their creation, much like the authors of a book or a song do.

But source code is hidden from the view of most software users, making it more difficult for the non technically savvy to recognise this value. And when software systems grow complex, the number of contributors grows too, making it difficult even for the technically savvy to give credit.

And yet, recognising the value and importance of the contributors, not only of the developers, but also of the creators that design the algorithms and the architecture behind the code, is important, as we need to attract the best minds to develop the software infrastructure of our society.

We are particularly worried about the impact of the lack of recognition in research. Software must be recognised as a research product, and software development as a research activity in academic evaluations and assessments. For example:

→ **Software as a research product**
researchers, funders, publishers and institutions have recognized the need for preserving software associated with research data, but there are very few notable incentives, and practically no standards and processes for the collection and preservation of research software. Consequently, a large part of scientific results cannot be reproduced nor verified, due to the unavailability of the software artifacts that were instrumental in obtaining them.

→ **Software development as a research activity**
in many countries, software development is not taken into account when evaluating and promoting academics; as a consequence, there is less of an incentive for researchers to engage in the development and sharing of sophisticated software systems, thus society is depriving itself of significant contributions from researchers to the evolution of software development and technologies.

3.4 Lack of recognition for women and underrepresented communities

In the history of computing, women like Grace Hopper, Margaret Hamilton or Henriette Avram have made important contributions, and yet there is still a significant lack of recognition of the importance of women. They were often hidden figures in the history of the digital revolution.[28] This has serious consequences: role models are important in attracting new generations to computing, and to nurture an environment where all genders are welcome. As such, women's contributions should be respected and recognised.

Gender is, of course, not the only reason why certain contributions have been underappreciated. People from many types of groups traditionally underrepresented in computing, from geographic minorities to people with disabilities have made positive change in the world through computing. For example, in Sub-Saharan Africa despite limited resources and infrastructure, there have been innovative approaches to enable citizen engagement (e.g. Ushahidi)[29] or allow the unbanked[30] in these regions to easily save and transfer money. A holistic approach to sharing and preserving software source code allows recognition of these advances as well as those that are more traditionally accounted for in computing history.

3.5 Lack of incentives to release (legacy) source code

Managing the life-cycle of software products is a complex and costly process, and there is little incentive today to plan for the proper release of the source code of non-public and proprietary software for example when it is no longer commercially viable. As a consequence, recovering important technological milestones of the history of computing may end up requiring a huge amount of effort: the recovery of the source code of MS-DOS, Eudora or the legendary Xerox Alto are the result of long term dedication to overcoming both technical and legal hurdles.

Source code of legacy and contemporary software should be considered part of the world's heritage and is essential for education on best practices and reflection on the evolution and history of the computing profession. A concerted effort to collect, document, and make publicly available the source code of software is important to ensure the ability to modify, redesign and reproduce software of similar functionality in the future.

But these after-the-fact efforts may not be enough. The release of previously private source code by a commercial entity can be a substantial undertaking. It entails legal due diligence including identification of authorship and ownership, checking for disclosure of trade secrets including from suppliers, checking for trademark infringements, checking for patent coverage and much more. Without incentives of comparable value,

for-profit entities are unlikely to release legacy source code, especially code that has been "orphaned" by corporate mergers and acquisitions. Suitable incentives might include tax credits for the cost of the work involved and partially for the value of the code disclosed.

3.6 Lack of an universal catalog

Source code is spread around a variety of platforms and infrastructures that we use to develop and/or distribute it, and software projects often migrate from one to another. Millions of projects are developed on publicly accessible code hosting platforms, such as GitHub, GitLab.com, SourceForge, Bitbucket, etc., not to mention the myriad of institutional or community "forges" scattered across the globe, or developers simply offering source code downloads from their web pages. Projects tend to move between code hosting places during their lifetime, following current trends or the changing needs and habits of their developer communities. And the same source code can be copied in different places for the purpose of distributing it. As a consequence, today it is very difficult to explore the software commons as a whole.

And unlike what happens for books, music or other parts of our cultural heritage, we do not have a clearly defined way of naming, referencing[31] and citing[32] software projects, especially modern software systems developed collaboratively.

We lack a universal, comprehensive catalog of all the source code. This makes it more difficult to preserve or provide access to historically important software source code.

3.7 Lack of an universal repository

As with all digital information, software source code can be deleted, corrupted or misplaced. Developers have long relied on code hosting platforms to take care of their code, and keep track of all the versions of its development history.

While these platforms are indeed tools to enable collaboration and record changes, none of them offers any long term access guarantees: digital content stored there can be altered or deleted over time. In recent years we have seen major code hosting platforms shut down[33], endangering hundreds of thousands of publicly available software projects at once.

Long term preservation cannot be assumed by entities that do not make it a stated priority: preservation may be a side effect of other missions, but non-preservation focused institutions have other priorities that in the long term usually interfere with serving as a comprehensive repository.

We lack a *universal, comprehensive and long-term* repository that is dedicated to ensuring that if source code disappears from a given code hosting platform, or if the platform itself disappears altogether, the code will not be lost forever.

3.8 Lack of large scale open research infrastructure

With the growing relevance of software, it is increasingly more important to provide the means to improve its quality, safety, and security. This requires access to the full corresponding source code.

Modern advances in scientific research, in artificial intelligence, big data, static analysis, and many others areas, could be used to analyse the whole body of publicly available source code, finding errors, and fixing them, providing recommendations and speeding up innovation in many ways if we only had a place where all information about software projects, their public source code, and their development history is made available in a uniform data model. Some building blocks of this infrastructure are starting to emerge: Software Heritage provides a long term archive that offers a uniform representation of software source code, and its development history, across all code hosting platforms and version control systems[34]; various projects keep track of the activity on specific code hosting platforms, like GitHub[35] or SourceForge [36]; others record developers exchanges on specific platforms like StackOverflow or mailing lists.[37]

We need to blend all these efforts into a "very large telescope" of source code—in the spirit of the great mutualized research infrastructures such as the Very Large Telescope in the Atacama Desert or the Large Hadron Collider in Geneva; we need to make it available to developers, engineers, research bodies and industries in order to improve the software on the quality of which our society depends.

3.9 A clear and present danger: losing the earlier creators

Unlike many other scientific and technical disciplines, software is very young: the first real programs were written little more than half a century ago. Many of the minds that laid the foundations of the modern digital revolution are still alive, and willing to contribute their knowledge and their personal collections of software source code to build the history of software.

This is a unique opportunity for collecting our cultural heritage, and keeping track of the exceptional history of software technology and computer science[38].

Taking action today is more urgent than ever, as we have only a few years left: every year that goes by, we see some of the important figures in this landscape passing away; and the physical media on which they stored their source code decay too.

3.10 Dispersion of efforts

The threats identified in this report are not new, and have been observed and described repeatedly before, with several initiatives trying to address these issues in a variety of ways.

Many focus on high quality curation of selected material, that may be already in, or donated to, a museum, an archive or a research library.

Others focus on curating metadata for identifying software, be it in crowdsourced efforts like Wikidata, or in industry driven efforts to improve software traceability.

Still others look for ways of citing, referencing or archiving selected portions of the software commons, most notably scientific software.

Many of the issues faced by all these initiatives are common: collecting, referencing, citing, describing, archiving, preserving and making available the source code of software artifacts.

These issues must be addressed in a coordinated way, limiting the proliferation of different standards and focusing on making initiatives interoperable. Avoiding duplication of efforts would be ideal, but the worse scenario would be incompatibilities between independent initiatives such that it is impossible to ever make them work together. Having five copies of the same catalog to avoid data loss is a blessing, but having five different incompatible catalogs means we do not have a catalog at all.

To this end, proper communication should be fostered among these initiatives and actors, including through the establishment of an international working group, following the example of the Software Preservation Network and the Digital Preservation Coalition, that have gone to great length to bring together practitioners from multiple institutions in the broader scope of digital preservation.

4. Stakeholders

Since software is everywhere, there is a broad variety of stakeholders that have a role to play to address the threats that software source code faces today. We try to identify here a few broad classes of actors, their roles, their stakes, and the actions they can take.

4.1 Governments / Intergovernmental Organizations

As software users, commissioners and producers

Both governments and the public sector - at local, national and international levels - are among the largest software users, commissioners[39] and producers, hence some of their *policies* have a deep

impact on the software market. The imperatives of transparency, accountability and efficiency have led many governments to work together, in the Open Government Partnership, and to draft policies, like the «Open Source Contribution Policy», that *raise awareness of the importance of software source code, and recognise the effort of developers*.

As law makers
Governments and supranational bodies are important stakeholders: they have the power and the responsibility of formulating international instruments or drafting and passing laws that may have a great impact on the future of software source code. For example, software source code is very different in nature from traditional copyrighted material, like songs, books, or movies, and requires that the legal framework for copyright must be modified to take these specificities under account. Policy makers should ensure to get adequate information and counseling about software in order to craft policies that account for the unique needs of software infrastructure and that appropriately address the consequences of the use of software for public decision making.

4.2 Research and Higher Education
Universities and research bodies are directly concerned when it comes to research software source code, for many reasons, including producing software.

Open Science/Open Research and Reproducibility
The stated goal of Open Science is to set the default to open for research results («as open as possible, as closed as necessary»). Maximally open source code for research software would enable science to grow faster, allow today's researchers to "stand on the shoulders of giants" by sharing and extending the software written by those giants, and would also substantially improve the reproducibility of computational research across the disciplines.

To reach this goal, every body must play its role. Funders (private and public) should consider requiring the research they fund to make the resulting source code available for reuse under a standard license, and have it deposited in an archive for the long term. Learned societies and research governing bodies should elaborate and encourage code of conducts for researchers that value the software contributions. Publishers and Open Access archives should offer services to deposit research software source code alongside publications.

All these actions should be coordinated, and adopt standards for long term software source code archival: we must mutualise what is common, and not waste resources in a myriad incompatible initiatives.

Software development literacy Producing good software is not an easy task: it requires proper skills, that must be disseminated widely among researchers.

The transfer of knowledge and practices in using legacy digital technologies needs to be supported by the development of educational courses at all levels, and the recognition of professional profiles such as digital historians, that can study how to access source code, executable software, and documentation over time.

4.3 Education
In order to avoid another digital divide in the new generations, it is important to offer a curated and well documented collection of software (source code) that shows illustrative examples of software programs, similarly to the selected books that are offered for students to read in order to learn to use proficiently their own language. This teaching material needs to be carefully selected source code, to illustrate design across a range of different types of software and enterprises, together with design notes, algorithms and their implementation as source code on different software platforms and narratives of software development, especially collaborative endeavours. Documentation of design decisions and decisions regarding selection of implementation environments would also support educational application. Furthermore, this teaching material should be made available to all as Open Educational Resources.

4.4 Industry and Commerce

All sectors of modern industry and commerce use and/or produce software, and the amount of free and open source software that is reused is booming. There is a clear common interest in curating the software components that are openly shared and reused, and in ensuring that they are properly archived and documented for later reuse.

On the other hand, there seem to be few incentives for industries to release the source code and documentation of their own proprietary software once it becomes legacy. Awareness needs to be raised on the importance of not losing this legacy, which is a part of the history of computer science and technology.

Corporate actors also play a key role in lobbying for aspects of legislation that regulate their business and set the context for their long-term strategies, which often implicate non-corporate software sharing and use. It is thus vitally important that companies refrain from pursuing policy goals that protect their own interests at the cost of making it impossible to preserve software for coming generations.

4.5 Memory Institutions

Information preservation is the vehicle by which we capitalize the knowledge that humankind has built over our history, and advance our understanding of ourselves and our environment. The emergence of digital technologies has led to the rise of digital preservation. Software in general, and software source code in particular, must be considered an essential component in this endeavour, and treated as a priority, as the Software Preservation Network is doing. Memory institutions (such as galleries, libraries, archives and museums) should share efforts in identifying and tackling the challenges, and leverage existing infrastructures for source code preservation, whenever possible.

From the perspective of software creators, it becomes important to isolate software code as documentary heritage, oftentimes digitally born, which deserves to be preserved. However, in some cases software developers still write or print out code on paper which, in itself, may constitute significant heritage as a physical 'carrier' of the memories of software developers. Preserving these forms of source code is also particularly important in regions where computers may be inaccessible and teaching of computer programming occurs in the old 'analogue' format of paper documentation.

4.6 NGOs / Other standards-setting bodies

Many organisations have been working for years or decades to advocate, professionalise, and disseminate the principles behind the free and open source software that constitutes today our software commons. They have precious expertise to share with all the other stakeholders, and should be involved in their efforts.

Non-Governmental Organizations involved in the definition of standards which are most relevant to software development and preservation should also take part in these efforts.

4.7 Individuals

One oft-forgotten set of stakeholders for preservation and accessibility of software source code are individuals - citizens and enthusiasts of all types, from software development to computer history. Given the enormous amount of source code being produced, individuals play an important role in saving and documenting source code, in addition to being a target audience for software heritage efforts. Efforts to preserve and make software source code accessible should be open to the public and to the extent possible, involve members of the communities that developed the software or are impacted by it.

5. Glossary

Archive
An archive is a collection of primary source documents which are, for a variety of purposes, preserved over time (see https://en.wikipedia.org/wiki/Archive). In a software archive we intend to find of course the source code of the software, but also documentation, and other traces of the design and development process of software artifacts.

Catalog
By analogy with a library catalog (see https://en.wikipedia.org/wiki/Library_catalog) we mean here a register of all the software source codes found in a repository, a forge or group of them.

Forge, code hosting platform
A platform, usually web-based, for both developing and sharing software source code, see also https://en.wikipedia.org/wiki/Forge_(software)

Proprietary software
A software is called "proprietary", when it is not "open source" or "free software"; in particular, most proprietary software keeps its source code behind closed doors, see also https://en.wikipedia.org/wiki/Proprietary_software

Repository
A storage location in which software artifacts can be deposited, and from which they can be retrieved.

Source code
Common abbreviation of "software source code", it refers to the representation of a computer program that is best suited for a developer to make modifications to it; usually this is a (set of) textual document(s) written in a high level programming language.

Universal, comprehensive
A "comprehensive" catalog strives to cover a range as broad as possible of items. The attribute "universal" stresses the ability to address all intended needs, including interoperability.

END NOTES

1 http://www.un.org/ga/search/view_doc.asp?symbol=A/RES/70/1&Lang=E

2 D. boyd, «It's Complicated: The Social Lives of Networked Teens», 2015, Yale University Press

3 J. E. Hannay, C. MacLeod, J. Singer, H. P. Langtangen, D. Pfahl and G. Wilson, "How do scientists develop and use scientific software?", 2009 ICSE Workshop on Software Engineering for Computational Science and Engineering, Vancouver, BC, 2009, pp. 1-8. doi: 10.1109/SECSE.2009.5069155

4 Even if it is out of the scope of this document, preservation of software executables is a very valuable objective, and there are many ongoing efforts to address it.

5 https://www.gnu.org/licenses/gpl.html#section1

6 Preface to Abelson, Sussman, and Sussman, "The Structure and Interpretation of Computer Programs", MIT Press, 1985.

7 First page of Donald E. Knuth. 1984. Literate programming. Comput. J. 27, 2 (May 1984), 97-111, http://dx.doi.org/10.1093/comjnl/27.2.97

8 Shustek, L. J. "What Should We Collect to Preserve the History of Software?", IEEE Annals of the History of Computing, 2006.

9 Nancy Kranich and Jorge Reina Schement. Information commons. Annual Review of Information Science and Technology, 42(1):546–591, 2008.

10 See also https://en.wikipedia.org/wiki/Information_commons#Software_commons.

11 https://en.wikipedia.org/wiki/Free_and_open-source_software

12 See https://it-cisq.org/wp-content/uploads/2012/09/CISQ_2009_Executive_Forums_Report.pdf. See also the Reproducible Builds Project, https://reproducible-builds.org/

13 See the Open Government Declaration, https://www.opengovpartnership.org/open-government-declaration

14 Laurence Lessig, "Code is Law: on liberty in cyberspace", Harvard Magazine, January 2000, https://harvardmagazine.com/2000/01/code-is-law-html

15 "When algorithms affect human rights, public values or public decision-making we need oversight and transparency" Marietje Schaake, MEP

16 Informatics Europe & ACM Europe Council, "When Computers Decide: European Recommendations on Machine-Learned Automated Decision Making", February 2018. https://dl.acm.org/citation.cfm?id=3185595

17 For a more detailed outline of principles for accoutanble algorithmic systems, see http://www.fatml.org/resources/principles-for-accountable-algorithms.

18 Weitere Beteiligte (Hrsg. etc.): Alice Allen et al. Engineering Academic Software (Dagstuhl Perspectives Workshop 16252) doi: 10.4230/DagMan.6.1.1

19 R. J. LeVeque, I. M. Mitchell and V. Stodden, "Reproducible research for scientific computing: Tools and strategies for changing the culture," in Computing in Science & Engineering, vol. 14, no. 4, pp. 13-17, July-Aug. 2012. doi: 10.1109/MCSE.2012.38

20 See "Declaration of Principles", WSIS-03/GENEVA/DOC/4-E, World Summit on the Information Society, Geneva, December 12, 2003

21 CS for All: https://www.nsf.gov/news/special_reports/csed/

22 Informatics for All: https://dl.acm.org/citation.cfm?id=3185594

23 See the report https://globalyoungacademy.net/wp-content/uploads/2018/03/18013_GYA_Report_GARS-Web.pdf

24 https://blog.mozilla.org/netpolicy/2018/09/07/eu-copyright-reform-the-facts/

25 See https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study

26 See https://obamawhitehouse.archives.gov/blog/2016/01/30/computer-science-all and https://www.axios.com/trump-to-announce-computer-science-education-initative-1513305746-24c9b0c5-b6e6-4bae-9a85-425ac65e8f24.html

27 In the United States, there is documentation that suggests that legal risk and a "permissions culture" has led to extremely limited collection of software works. See Patricia Aufderheide, Brandon Butler, Krista Cox, and Peter Jaszi1, The Copyright Permissions Culture in Software Preservation and Its Implications for the Cultural Record, https://www.arl.org/storage/documents/2018.02.09_CopyrightPermissionsCulture.pdf.

28 See Hidden Figures: The Story of the African American Women Who Helped Win the Space Race by Margot Lee Shetterly.

29 See https://www.ushahidi.com/about Last accessed: 06 November 2018

30 Dr. Lennard Bangens and Bjorn Soderberg, 'Mobile Banking- Financial Services for the Unbanked' (SPIDER: The Swedish Program for ICT in Developing Regions, 2008 https://spider1.blogs.dsv.su.se/wp-content/blogs.dir/362/files/2016/11/Spider-ICT4D-Series-2-Mobile-banking-financial-services-for-the-unbanked.pdf Last accessed: 06 November 2018. See page 4 for a definition of unbanked

31 Roberto Di Cosmo, Morane Gruenpeter, Stefano Zacchiroli. Identifiers for Digital Objects: the Case of Software Source Code Preservation. iPRES 2018 - 15th International Conference on Digital Preservation, Sep 2018, Boston, United States. pp.1-9, https://hal.archives-ouvertes.fr/hal-0186579

32 Smith AM, Katz DS, Niemeyer KE, FORCE11 Software Citation Working Group. (2016) Software citation principles. PeerJ Computer Science 2:e86 https://doi.org/10.7717/peerj-cs.86

33 Most notably, Gitorious (2015) and Google Code (2015)

34 Jean-François Abramatic, Roberto Di Cosmo, Stefano Zacchiroli: Building the universal archive of source code. Commun. ACM 61(10): 29-31 (2018)

35 See GHTorrent, http://ghtorrent.org/, and GHArchive, https://www.gharchive.org/.

36 SourceForge research data, https://www3.nd.edu/~oss/Data/data.html.

37 See for example The Mail Archive https://www.mail-archive.com/, and the now abandoned Gmane.

38 Personal interviews of early creators are being recorded at the initiative of several memory institutions, but we would like to see also recordings of the actual process used by them to develop their software.

39 See for example Mariana Mazzucato, "The Entrepreneurial State", Anthem Press 2013

"In memory of Dr Indrajit Banerjee,
former Director of the Knowledge Societies Division,
Communication and Information Sector, UNESCO."

United Nations Educational, Scientific and Cultural Organization

Memory of the World

inventors for the digital world

Part of our Heritage, Pillar of our Present, Enabler of our Future.

More information

https://en.unesco.org/themes/building-knowledge-societies

softwareheritage_info@unesco.org