iPRES 2024 • iPRES 2024 Proceedings

Preserving Inria's Legacy Software: A Crowd-Sourced Approach

Mathilde Fichen[1] Roberto DiCosmo[2] Gérard Giraudon[3]

[1]Software Heritage, CNAM, [2]Software Heritage, [3]Inria Alumni

Published on: Aug 29, 2024

URL: https://ipres2024.pubpub.org/pub/hdap1420

Abstract - The present article discusses the efforts initiated by Software Heritage and the Inria Alumni Network to carry out a large-scale identification and preservation effort of the historical software artifacts created by Inria, the French National Institute for Research in Digital Science and Technology. In an attempt to scale up a work-intensive process, the authors have turned to a crowdsourced approach, with the first step consisting of better identifying Inria's software heritage.

More specifically, the article discusses the outcomes of a survey sent to current and past Inria employees. Its aim was to identify landmark legacy software products of the institute, available source codes, and possible contributors to a future collective archiving effort.

Unsurprisingly, the survey found that older software pieces are less likely to be identified, with only a small subset of early-age Inria software production being reported in the survey. The availability of associated source codes also varies, with older software being more likely to have partially or completely lost source code, or to be stored on obsolescent media. Source code from recent software however is more likely to be available on modern collaborative platforms like Github or Gitlab, ensuring safe archiving via the Software Heritage universal archive.

In conclusion, the Inria survey confirms the urgent need to identify and preserve historical software items, especially those from the earlier decades which are at higher risk to be definitively lost if no preservation effort is made. We believe that, due to the broad scale and breadth of this effort, the lessons learned will be of interest to other institutions and software preservation initiatives.

As a next step, we plan to set up a dedicated framework to empower Inria alumni to contribute to the preservation of the legacy codes, and we call on other institutions to join in this effort, paving the way for many other crowd-sourcing source code preservation initiatives.

Keywords – software preservation, legacy software, source code, crowd-sourcing

This paper was submitted for the iPRES2024 conference on March 17, 2024 and reviewed by Chris Prom, Genevieve Havemeyer-King, Dr. Lee Pretlove and 1 anonymous reviewer. The paper was accepted with reviewer suggestions on May 6, 2024 by co-chairs Heather Moulaison-Sandy (University of Missouri), Jean-Yves Le Meur (CERN) and Julie M. Birkholz (Ghent University & KBR) on behalf of the iPRES2024 Program Committee.

Introduction

If the interest and urgency of preserving our software heritage are identified by numerous individuals, associations, or institutions, the practical implementation of such archiving and conservation efforts remains

nonetheless complex. Various initiatives, addressing different objectives and adopting varied strategies, have emerged since the 1980s.

In 1988, the Library of Congress established a Machine-Readable Collections Reading Room [1] providing access to software and files that could be executed or read on historical machines. The project was abandoned after a few years due to its high cost and low number of users.

In 2002, Grady Booch, renowned computer scientist and software engineer associated with IBM, circulated an email titled "Preserving classic software products" in which he called for the identification of software that should be a priority for preservation efforts and for locating the source code [2]. Following this, in 2003, the volunteer committee initially known as the Software Collection Committee (now Software Preservation Group) [3] was established to support the Computer History Museum in Mountain View in its mission to preserve software.

In 2016, Software Heritage, a non-profit, multi-stakeholder initiative, was launched by Inria in partnership with UNESCO [4], aiming for the extensive and automated archiving of all publicly available source codes on software forges. This project primarily focuses on archiving recent source codes, and as of February 2024, more than 18 billion source files have been archived. In 2019, the Paris Call on Software Source Code as Heritage for Sustainable Development [4] emphasized the need to preserve source code as part of our human heritage.

Archiving legacy code can’t, however, be automated, due to the variety of formats and physical media - from paper listings to floppy discs - in which they are stored. A dedicated process was set up in 2021 by Software Heritage and the University of Pisa to handle the archiving of legacy source code, including proper versioning and metadata, the Software Heritage Acquisition Process (SWHAP) [5]. The SWHAP, as of now, has allowed experimenting with archiving landmark legacy source codes from the University of Pisa and from Inria, and a presentation of the archived materials can be found on the Software Stories platform [6]. The process is, however, work-intensive, and we believe that scaling up can only be achieved by a collective effort of the community.

In May 2023, Software Heritage and the Inria Alumni Network initiated a large-scale preservation effort geared towards Inria legacy software. The chosen approach was to start a massive crowd-source campaign through a survey that has been sent to current and former Inria employees and collaborators in order to balance the difficult equation of large-scale preservation with limited financial and human resources. To this end, support was sought and obtained from Inria leadership.

We believe Inria to be the ideal experimental ground for such an initiative. The French National Institute for Research in Computer Science and Control Theory was founded in 1967 as part of a broader national effort launched by Charles de Gaulle to accelerate the development of computer science and computing infrastructure

in France. With a focus on both theoretical research and practical applications, Inria has been involved in numerous projects that significantly contributed to the field. Notable historical initiatives include the Micado mission (1970) for computer-aided design and drawing [7], and the Cyclades project, carried out in the 1970s to explore innovative solutions for networking computers, resulting in the development of packet switching for data transfers that laid the foundations for the Internet [7][8]. Another example is the COQ environment (a formal proof management system), which was launched in 1984 and has continued to evolve with successive versions. At present, the Inria software library lists almost 1500 active references [9], some of which are used by vast communities of users within dedicated consortia. The total number of software products developed at Inria since its creation can be estimated at several thousand units [10]. Alongside the historical anchoring of the institute, its active alumni network, and its proximity to Software Heritage, seemed key advantages to undertake such an initiative.

As a first step, a survey was sent to past and present Inria employees, totally around 6000 people. The content of the survey can be found in appendix. The goal of the survey was three-fold: (1) identify Inria legacy software products (2) identify available source code and other relevant historical materials (3) identify volunteers to further help in archiving source codes and other materials in the Software Heritage archive. By software, we meant any computer code intended to be executed on a computer.

Our survey received 88 answers mentioning 109 software entries. In the following paragraphs, we provide an overview of the key insights we have derived from it and hope to engage with the broader software preservation community in planning the next steps.

What Is Legacy Anyway?

Defining ‘legacy’ in the context of software products is a complex task, and determining a cut-off date for what qualifies as a legacy software item appears arbitrary. Given the youthfulness of computer science as a discipline, we moreover lack the perspective to predict which types of historical software will remain relevant to future generations.

Consequently, we adopted an inclusive approach, allowing respondents to freely submit any software item they deemed pertinent to our survey, accompanied by justifications for considering it legacy. Despite anticipating predominantly ‘old’ software submissions, we were surprised to observe entries ranging from 1968 to 2022.

The main criteria used to justify legacy (35% of answers) was some sort of adoption rate or proof of success, such as: "*Big industrial and economic impact*", or "*Geometric library used worldwide*."

At the other end of the spectrum, and surprisingly, only three software pieces referred explicitly to some kind of age or vintage criteria:

"*This software, written in Algol W, was developed and run on an IBM card-programmed machine in a closet in Building 8 of Rocquencourt, connected via a 150 baud link to the CP-CMS of the IBM 360-90 at IMAG!*"

In between, we identified the following categories:

- Being part of a larger story (32% of answers): "*Skype ancestor*", "*This software is the predecessor of Centaur, which was one of Inria's flagship software programs in the 1990s*"
- Being the first of a kind (25% of answers): "*This interface is one of the first to utilize multi-font and multicolor display to enhance the readability of formal proofs*.", "*The first mesh generator to reach one billion elements*"
- Achieving some sort of scientific recognition (22% of answers): "*This algorithm was refined by numerous researchers. It was crucial for calculations involving the basis of unifiers in the commutative-associative case*", "*A software that had immense significance in automated theorem proving and equational computation*."
- Some sort of intrinsic quality of beauty of the software (10% answers): "*It was a well-crafted software written in Caml*.", "*Beautiful software*", "*A gem of programming in the Perl language*".

Note that several justifications could be given for a given software entry (ex. *part of a larger story* and *the first of a kind*).

In our opinion, the various justifications provided for considering software as legacy underscore the challenges of managing relatively recent legacy, such as software source code. We advocate for an inclusive approach, extending preservation efforts to any software deemed significant by the community, as setting a cut-off date would appear arbitrary in this context.

Identifying Inria Legacy Software

The older the rarer

Unsurprisingly, older software pieces are less likely to be identified: only one software item was submitted before 1970, and nine software entries were submitted for the 1970s decade, coming from only four respondents. These submissions represent only a small subset of Inria's (back then IRIA) early-age software production. Moving forward, the numbers increase with each subsequent decade: 19 software pieces were submitted for the 1980s, 22 for the 1990s, 24 for the 2000s, and 26 for the 2010s.

The oldest submitted software is the "LP10070 compiler" developed during the Esope project. Esope (Exploitation Simultanée d'un Ordinateur et de ses Périphériques) was a project to build a time-sharing exploitation system for the CII 10070 computer, that took place between 1968 and 1972 at IRIA [11]. LP10070 was the assembly language developed for the project. The source code of the LP10070 compiler is unfortunately lost.

Registered software pieces of the 1970s feature:

- The text editor of the ESOP project (1972)
- The STST protocol of the Cyclades project (1972). The Cyclades network was one of the pioneering works experimenting with the concept of packet switching, that led to the Internet
- UNIF (1974) a unification algorithm in lambda calculus using Church's type theory
- MODULEF (1974) is a general-purpose finite element library developed by the MODULEF club. The MODULEF club was founded by INRIA in 1974 with the aim of bringing together universities and industry, both French and foreign, in order to design and implement an extensive library of finite element modules.
- Mini Logic for Computable Functions (1974) interactive automated theorem prover in Algol
- Mentor (1975) a structure-oriented program editor, which served as a first step towards Centaur, the generic programming interactive environment generator
- Inrimage (1978) an image processing software, later commercialized by the company NOESIS.
- BASIS (1978) an algorithm to generate the basis of solutions to homogeneous linear diophantine equations

As time passes, identifying the oldest software items and tracing related source code becomes increasingly difficult. This task heavily relies on a small group of active Inria alumni dedicated to preserving the institute's heritage. The survey underscores the urgency of undertaking preservation actions while historical contributors are still actively involved.

A family affair

Moving forward in time, it is interesting to see that some of the registered software pieces reference one another, thus creating some sort of lineage between them.

Centaur (1985) is an interactive programming environment that stemmed from the Mentor project (1975). The Centaur capabilities were used in CtCoq (Ct stands for Centaur), an interactive interface for the Coq theorem prover (1992)

CONSTR (1984) later developed as Coq (1989), the groundbreaking proof assistant that received the ACM software system award in 2013 [10]. CtCoq (1992) became the main interface for Coq, introducing original features such as different police sizes and colors to help with reading formal proofs. CtCoq evolved in PCoq (1998) to become independent of LeLisp language, whose future was becoming uncertain.

Le ML (1980) is the first implementation of the ML language developed at Inria. CAML (1985) is the second implementation. It was used to implement the initial versions of the famous Coq proof assistant. A precursor to Caml Light (1990) and the renowned OCaml programming language (1990) that received the ACM Programming language award in 2023 [12].

Le Lisp (1980) was Inria implementation of the Lisp language. ObjVLisp (1984) is an object-oriented extension of the language.

FreeFem (1987) is a popular 2D and 3D partial differential equations solver still used by thousands of researchers across the world. FreeFem was rewritten in C++ and became FreeFem++ (1999)

The lineage between some of the submitted software pieces calls for a specific effort in archiving the related source codes to preserve the connection between them. This raises however technical and infrastructure challenges that may require to adapt our current SWHAP process.

Identifying source code archives

Apart from identifying legacy software of interest, we also asked respondents about the software source code and other potential archived documents.

The source codes of software created in the early years of Inria are more likely to be lost, or at least to remain unidentified for now. Out of 9 registered software from the 70's only 3 have identified source code archives (all detained by one of our dedicated Inria Alumni under the form of many kilograms of paper listings). From 19 software pieces identified from the 80's, 12 have fully preserved source code archives.

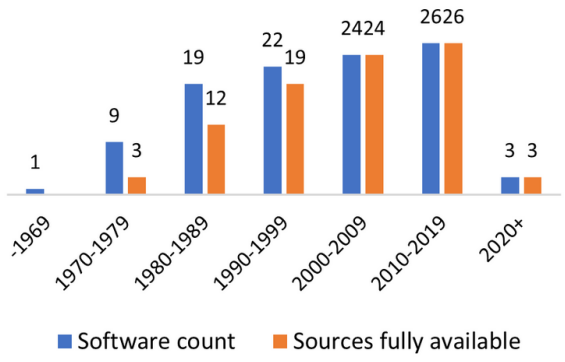

Fig.1 Count of submitted software products per decade and count of those whose source code is fully available

The chart illustrates that older software is more likely to have partially or completely lost source code. Additionally, the quality of source code archives format improves with more recent software.

In terms of proportions, 16% of the submitted software has either partially or completely lost source code, 5% have source code available in paper listings, 40% exist in some digital format including compact discs or private hard drives, and 38% are on modern forges. Only the latter are currently automatically archived on Software Heritage universal archive. This means that 62% of the referenced source code is either lost or at risk of being lost if active archiving efforts are not undertaken.

Next Steps

In total, fifteen respondents, covering a total of thirty-two software items, indicated they were willing to dedicate some time to help with preserving the identified legacy source codes. As a next step we would like to organize working groups to implement proper archiving of the source codes and related materials in the Software Heritage universal archive, leveraging the existing Software Heritage Acquisition Process [5], designed to curate and archive legacy code.

If the number are still modest, we hope that the experiment will help us improve the SWHAP process [5], build a community of practice around legacy source code archiving and encourage other institutions to contribute to the archive.

Lessons learned

We retain the following lessons from this first step in our effort to recover all relevant legacy software at Inria, and we hope they can be of interest for similar other efforts at comparably large institutions or communities. The first observation is that the survey approach effectively identifies significant legacy software, engaging the community to uncover hidden digital gems, but to elicit sufficient response it is essential that it is backed institutionally: in our case, the Inria Alumni Network and the official message of Inria’s CEO played a key role. The responses to the survey confirmed that the involvement of individuals with firsthand knowledge is crucial for providing context and ensuring comprehensive preservation. It comes as no surprise that early software faces higher risk of loss, so proactive preservation and dedicated frameworks like SWHAP are essential for safeguarding digital artifacts. Finally, the crowdsourcing approach helps to build a community around preservation efforts facilitates knowledge sharing, resource pooling, and raises awareness of software’s cultural significance: in our case, the newly created dedicated mailing list already counts 59 members.

Conclusion

This survey underscores the urgent need to identify Inria's historical software, particularly those from the 1960s, 1970s, and 1980s, as our collective memory of them gradually fades. Additionally, a substantial portion of the source code developed before the 2000s lacks secure archiving, and we believe it is crucial to make efforts to preserve it for the benefit of future generations. We hope that the lessons learned from this survey can raise awareness, and we call now on all kindred spirits in the software preservation community to join forces and share knowledge and good practices while we move forward to establish a community of volunteers dedicated to preserving Inria's legacy software. We plan to initiate actions in this direction in 2024.

Appendix - Survey content

The survey was built using Framaform, a free software solution for building surveys, respectful of data privacy. You will find below the content of the survey.

Introduction

The Inria alumni network, Software Heritage, and the Direction of Culture and Scientific Information (DCIS) of Inria are inviting you to contribute to the inventory of the software heritage built at Inria since its inception.

This survey will allow you to share your knowledge and help create a detailed overview of the software that has contributed to the institute's history, as well as the teams and individuals involved in these software projects, and the artifacts that can still be preserved.

The survey will remain open until September 15th.

The results will be consolidated in an anonymized report that will be made public this autumn. This report will lay the foundation for the implementation of a policy to preserve the source codes and other archival documents related to the history of the identified softwares, in collaboration with Software Heritage and Expo Inria.

Estimated time: 5 to 10 minutes.

Information regarding the protection of personal data

In accordance with Article 13 of the Regulation on the protection of individuals with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation or GDPR), please find below some information regarding this data collection:

The representative of the data controller is the CEO of Inria.

The Data Protection Officer (DPO) appointed by Inria can be contacted via email at dpo @ inria.fr.

The purpose of this processing is for the instruction by the Inria Alumni and Software Heritage services.

The legal basis for this processing is consent.

Your data will be kept for a maximum duration of one year.

You have the right to access your data or request its erasure. You also have the right to object, rectify, and limit the processing of your data.

This data collection is voluntary. You are not obligated by any regulatory, contractual, or other means to provide your data.

This processing does not involve any profiling or automated decision-making concerning you based on your data and its processing.

By clicking Next, you acknowledge that you have read and understood these provisions.

Personal information

- Last name *
- First name *
- E-mail address

Inventory of Inria legacy software

By software, we mean any computer code intended to be executed on a computer.

In your opinion, which historical softwares from Inria would justify preservation efforts? You can consider softwares that have been innovative, achieved particular success, or reflect the broader history of computing.

You do not need to have contributed to these software projects to list them. If you would like to list more than five software, please submit the survey a second time.

General information about the software

- Software name
- Approximative date of software first version
- Link to website (if existing)
- Contact point(s)
- Last name, first name of main contributor(s)
- Responsible team(s) at Inria
- Scientific or technical significance of the software (in your opinion)
- Historical significance (in your opinion)
- Other information you would like to share about the software

Identifying historical artefacts

Did you contribute yourself to the creation of this software?

- Yes
- No

Have you (or someone you know) preserved some artefacts of relevance to the history of the software?

- Source codes
- Other personal records
- Format and location of the artefacts

For source code, please provide the format (paper listings, floppy disk, online files etc.) and the associated URL if existing. For other items, provide the type (technical reports, pictures, mails etc.)

A call for participation

As part of its collaboration with UNESCO, Software Heritage has developed a procedure called the Software Heritage Acquisition Process (SWHAP) aimed at collecting, curating, archiving, and presenting historical source code. Initial tests have been conducted, and you can find the resulting presentations here.

To improve this process and contribute to the preservation of Inria's software heritage, we are seeking volunteers who have kept historical source code and related artefacts and who could dedicate some time to their preservation.

We are planning an information meeting soon (exact date to be determined) to launch the collaboration with these testers.

Would you be interested in participating?

- Yes
- No

Merci !

The teams of Inria Alumni Network, Software Heritage, and DCIS thank you for your time!

Would you like to be kept informed of the survey results?

- Yes
- No

Would you like to join the Software Heritage project mailing list?

- Yes
- No

ACKNOWLEDGMENTS

We extend our sincere appreciation to all participants who contributed to the survey, providing valuable insights and information.

We would like to express our gratitude to UNESCO for their funding, which not only supported the development of this article but also played a crucial role in supporting the current initiative at Software Heritage regarding legacy source code preservation.

References

1. J. Kimball, S. Thorin, and L. Arret, “Providing Reference Assistance for Machine-Readable Materials:,” Ref. Libr., vol. 14, no. 31, pp. 31–38, Dec. 1990, doi: 10.1300/J120v14n31_04. ↩

2. P. McJones, “In Search of the Original Fortran Compiler,” IEEE Ann. Hist. Comput., vol. 39, no. 2, pp. 81–88, 2017, doi: 10.1109/MAHC.2017.20. ↩

3. “Software Preservation Group — Software Preservation Group.” Accessed: Oct. 18, 2023. [Online]. Available: https://www.softwarepreservation.org/ ↩

4. “Paris Call: Software Source Code as Heritage for Sustainable Development,” 2019. [Online]. Available: https://unesdoc.unesco.org/ark:/48223/pf0000366715.locale=fr ↩

5. L. Bussi, R. Di Cosmo, C. Montangero, and G. Scatena, “Preserving landmark legacy software with the Software Heritage Acquisition Process,” in iPres2021-17th International Conference on Digital Preservation, 2021. ↩

6. M. Gruenpeter, R. Di Cosmo, K. Thornton, K. Seals-Nutt, C. Montangero, and G. Scatena, “Software Stories for landmark legacy code,” Inria, 2021. ↩

7. A. Beltran and P. Griset, “Histoire d’un pionnier de l’informatique: 40 ans de recherche à l’Inria.” EDP Sciences, 2020. ↩

8. A. L. Russell and V. Schafer, “In the Shadow of ARPANET and Internet: Louis Pouzin and the Cyclades Network in the 1970s,” Technol. Cult., vol. 55, no. 4, pp. 880–907, 2014, doi: 10.1353/tech.2014.0096. ↩

9. “La production logicielle | Inria.” Accessed: Mar. 07, 2024. [Online]. Available: https://www.inria.fr/fr/la-production-logicielle ↩

10. “Welcome! | The Coq Proof Assistant.” Accessed: Mar. 07, 2024. [Online]. Available: https://coq.inria.fr/ ↩

11. C. Bétourné, J. Ferrié, C. Kaiser, S. Krakowiak, and J. Mossière, “ESOPE : Une étape de la Recherche Française en Systèmes d’Exploitation (1968-1972),” presented at the 7ème Colloque sur l’Histoire de l’Informatique et des Télécommunications, Nov. 2004, p. 173. Accessed: Mar. 07, 2024. [Online]. Available: https://hal-lirmm.ccsd.cnrs.fr/lirmm-00108869 ↩

12. “Programming Languages Software Award.” Accessed: Mar. 11, 2024. [Online]. Available: https://www.sigplan.org/Awards/Software/ ↩