



NCDD

NETHERLANDS
COALITION FOR
DIGITAL
PRESERVATION

DIGITAL HERITAGE NETWORK

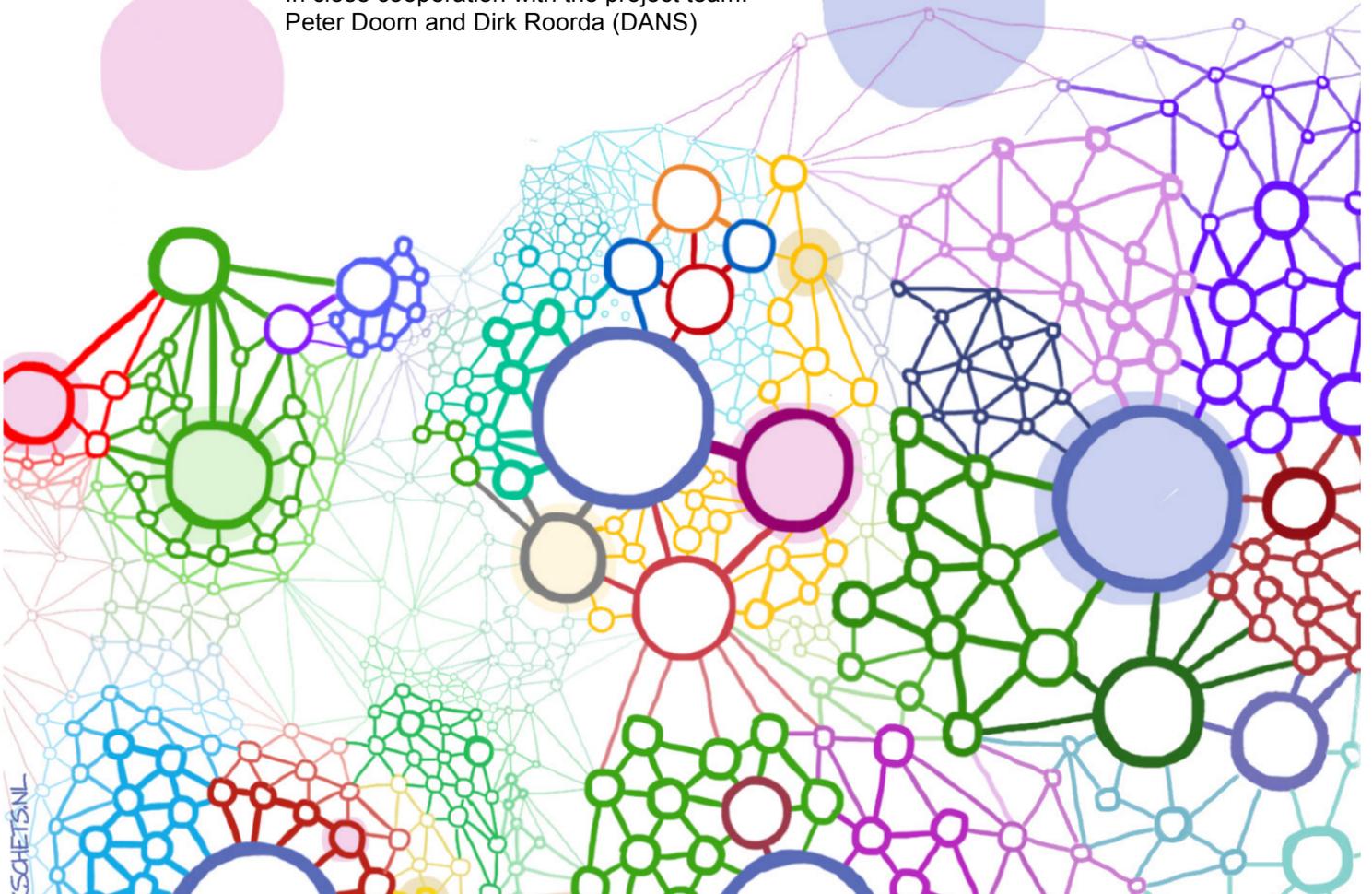
Software Sustainability

Final report

The Hague, 8 November 2016

Patrick J.C. Aerts
Data Archiving and Networked Services
patrick.aerts@dans.knaw.nl

In close cooperation with the project team:
Peter Doorn and Dirk Roorda (DANS)



Software Sustainability

Final report

The Hague, 8 November 2016

Patrick J.C. Aerts
Data Archiving and Networked Services
patrick.aerts@dans.knaw.nl

In close cooperation with the project team:
Peter Doorn and Dirk Roorda (DANS)

www.den.nl/netwerkdigitaalerfgoed
www.ncdd.nl
www.dans.knaw.nl

Acknowledgement

This report describes the research into the sustainability of software. The research was carried out by Data Archiving and Networked Services within and with funding of the [Dutch Digital Heritage Network](#). The national Digital Heritage Network was established in 2015. This network's strategy is based on three connected pillars: 1) increasing the visibility of collections, 2) improving the possibilities for using collections and 3) cross-sector sharing, utilisation, and scaling up of facilities for sustainable preservation and access. For each pillar, a work package was devised, consisting of a series of interlinked collaborative projects. The ultimate goal is to arrive at a future-proof and cost-efficient infrastructure that will encompass all of the various domains while meeting the needs of heritage users.

The Netherlands Coalition Digital Preservation was tasked with carrying out the work package "Sustainable Digital Heritage", that includes the study at hand. The aim of this section of the national strategy corresponds with NCDD's mission: to ensure the long-term accessibility of digital information through the establishment of a national network of facilities. This work package runs from September 2015 to May 2017.

1 Contents

1	Contents.....	4
2	Conclusions and recommendations	6
3	Introduction	9
4	Software sustainability in science and culture	11
4.1	Introduction	11
4.2	Science.....	12
4.3	Libraries.....	12
4.4	Archives.....	13
4.5	Broadcasting, Sound&Vision.....	13
4.6	Game preservation	14
4.7	Museums: Born digital art	14
4.8	Digital history and archaeology	15
4.9	Common issues across all domains	15
4.10	Legal matters	15
4.11	Bit rot and link rot.....	16
4.12	Lack of original devices	17
5	Need for a software sustainability facility.....	19
5.1	Introduction	19
5.2	Support from the science community	19
5.3	Past versus future	19
5.4	Expertise in an international context	19
6	Use cases	20
6.1	Introduction	20
6.2	General conclusions based on the use cases.....	20
6.3	Use case: Pineapple, a semantic search engine for historians	20
6.4	Use case: CATCH (Continuous Access To Cultural Heritage).....	22
6.5	Use case: Software for access to legacy CD-ROM collection	24
6.6	Use case: NDL (Naive Discriminative Learning), an implementation in R	26
6.7	Use case: eSTeP (eScience Technology Platform).....	28
6.8	Use case: Tundra, treebank repository and search tool	29
6.9	Use case: Gaming software	32
6.10	Use case: HSN (Historical Sample of the Netherlands).....	34
6.11	Use case: Preserving sound and vision	36
6.12	Use case: TSTC, software to manage Dutch language data	36
6.13	Use case: Digital archaeology – <i>De Digitale Stad</i>	39
6.14	Use case: Born digital art – SHFT-34.....	41
6.15	Observations from the use cases	42
7	International initiatives	43
7.1	Introduction	43
7.2	Knowledge Exchange group (KE)	43
7.3	UNESCO Persist	43
7.4	Digital Preservation Coalition	43
7.5	Software Heritage Project.....	43
7.6	PLAN-E	44
7.7	Commercial initiatives.....	44
7.8	UK	44
7.8.1	Centre for Digital Heritage, University of York	44
7.8.2	Digital Preservation Coalition	44
7.8.3	The Software Sustainability Institute (SSI)	44
7.8.4	Future-Proof Computing Group, University of Portsmouth	46
7.8.5	The Computer Conservation Society.....	46
7.9	Australian National Data Service (ANDS)	46
7.10	Denmark: Royal Library	47
7.10.1	DEFF, the Danish Electronic Research Library	47
7.10.2	Danish Royal Library.....	47
7.11	France	47
7.12	Germany.....	49
7.12.1	Technical University of Munich	49
7.12.2	University of Freiburg	49

7.12.3	Karlsruhe University	49
7.13	The Netherlands.....	49
7.13.1	SURF	49
7.13.2	NWO	49
7.13.3	DANS	50
7.13.4	NCDD and NDE	50
7.13.5	NLeSC	50
7.13.6	ePLAN	50
7.13.7	Legacy Coalitie NL	50
7.14	USA.....	51
7.14.1	CSESSP Workshop.....	51
7.14.2	Software Infrastructure for Sustained Innovation (SI2 - SSE&SSI).....	51
8	Software types	52
8.1	Semantics	52
8.2	Software classification.....	52
8.2.1	Firmware	53
8.2.2	Embedded software.....	53
8.2.3	System software	53
8.2.4	Application software	55
8.2.5	Internet applications.....	56
8.2.6	Mobile apps	57
8.3	Other characteristics axes	57
8.3.1	Open software versus closed (proprietary) software.....	57
8.3.2	Commercial software versus freeware.....	57
8.3.3	Licensed software versus unlicensed freeware.....	57
8.3.4	Off-the-shelf software versus tailor-made software.....	58
8.3.5	Individual programs versus software packages.....	58
8.3.6	Owned software versus software as a service.....	58
9	Appendix	59
10	Abbreviations and terms used.....	61
11	Article from <i>De Telegraaf</i> (in Dutch)	71

2 Conclusions and recommendations

1. Due to the inaccessibility of digital born¹ information, exponentially increasing volumes of contemporary and recent history are at direct risk, including news, trade, knowledge and methodologies. This applies to all domains covered by the Digital Heritage Network and NCDD.
 - ☞ Encourage and enable national actions to be taken towards software recovery and future sustainability:
 - ◆ From education and training to dedicated services (such as a Software Sustainability Initiative);
 - ◆ Continue with further research into the identified issues and continue to monitor international efforts and developments;
 - ◆ Allocate a leading role to a national coalition, involving NDE/NCDD partners, NLeSC, SURF and ePLAN;
 - ◆ Mobilise actors and bridge gaps between formal parties/actors and volunteers/amateurs.
2. Software Sustainability is lagging behind compared to Data Stewardship policies.
 - ☞ Treat Software Sustainability and Data Stewardship on an equal footing, policy-wise:
 - ◆ Software and Data are intrinsically closely connected and so are their urgent preservation needs;
 - ◆ In practical matters, however, Software Sustainability requires different solutions.
3. Split the matter into two parts: legacy and future. Since the amount of legacy software in all the domains of science, culture and heritage will keep on growing beyond the limits of what can be dealt with in a reasonable manner, it helps to attack the issues from two sides.
 - ☞ Invest in serious efforts to assign a proper place to software sustainability (and data stewardship) in education:
 - ◆ Include it in first and second-year curricula of universities, colleges for higher vocational education and other educational institutions;
 - ◆ Compile and share courses and training for students and researchers in the field of software development, both professionals and amateurs;
 - ◆ Consider sharing expertise in software building and maintenance through a network of Software Sustainability Initiatives across Europe;
 - ◆ For the Netherlands: appoint parties to materialise such a Software Sustainability Initiative;
 - ◆ Consider extending the present scope of Software Sustainability Initiatives to the library, archiving, gaming and museum domains.
4. The use cases show both common elements and domain-specific particularities.
 - i) There clearly are common elements, faced by all parties:
 - ☞ Cooperate wherever you can and differentiate where needed;
 - ☞ Cooperate to meet all common challenges and acquire critical mass for addressing the issues;
 - ☞ Further explore the common issues for common solutions.
 - ii) Very specific elements that differ across stakeholders:
 - ☞ Concentrate on the particularities as an addition to the common elements;
 - ☞ Share the specifics with peer communities elsewhere.

COMMON

5. Common to the Software Sustainability efforts in all domains are the legal matters. Today's legislation complicates preservation. Even if one technically succeeds in preserving a piece of software, the way in which this has to be done may violate copyrights and licensing legislation. This is certainly true for the actual deployment of the software. On the other hand, if it is not preserved now, the product may disappear entirely.

¹ The term "digital born" essentially refers to material that was not originally published on paper or never existed on paper, but was directly published in a digital format.

- ☞ Consider changing legislation (preferably at the European level for European countries) in order to allow national institutes with a legal task concerning (cultural) heritage to copy and use software and data for specific purposes.
 - ☞ Specific attention should be paid to:
 - ◆ Copyrights on orphaned or otherwise abandoned software, including operating systems and libraries;
 - ◆ Copyrights in conjunction with public access to Web archives;
 - ◆ Copy-protection infringements.
 - ☞ Cooperate with and support the UNESCO PERSIST project, which is presently working on an agreement with major global software vendors on the use of their obsolete products.
 - ☞ Consider adopting legislation against the present freedom of major, usually global commercial archiving services to discard data (photos and videos, websites, scientific data, resources and software (git)) without proper notice or solid alternatives.
 - ☞ Design criteria for keeping software (and maintaining it at a certain level) and for discarding it.
6. Common to the archival, library and museum domains is the issue of hardware availability. It applies to the science domain too, but to a much lesser extent. The problem arises from the many different storage devices (tapes, magnetic and optical disks and other I/O peripherals) that have been in use that still contain unique material, but require original systems to be accessed. Also, older computers may be necessary to run older software. There are still working systems around, but they are often in private collections with the serious danger of falling apart.
- ☞ Consider making a national (followed by a European) inventory of working computer and peripheral equipment; an informal inventory of sites with systems (working and not working) already exists.
 - ☞ Consider setting up a national computer museum with sufficient critical mass to have impact. A foundation has been laid in Zwolle, but it covers only a fraction of what would make a collection with a minimum level of completeness.
 - ☞ Start cooperating with other sites (including museums) and communities maintaining computer equipment (UK, Germany, France, Australia, US).
 - ☞ Adopt large-scale emulation as the carrier of software preservation.
7. Obviously, bit rot adds to the sense of urgency when considering software sustainability, but the technical issues related to bit rot are the same issues that pertain to data and data stewardship. Because software extends the domain of involvement to the Internet and the World Wide Web, link rot adds to the problem.
- ☞ Gain knowledge of bit rot matters from the data world.
 - ☞ Support existing national and international efforts towards:
 - ◆ Internet archives;
 - ◆ Persistent Identifiers (PIDs);
 - ◆ Persistent Uniform Resource Locators (PURLs).

SPECIFIC

8. Software Sustainability has different goals in different NDE / NCDD stakeholders' domains. Cultural heritage plays a role across all domains but with different intensities.
- In the museum/art sector the most important element is to keep acquired artefacts alive in such a way that the observer gets the impressions intended by the artist. The required software is only part of the total issue.
 - ☞ Consider investments in software sustainability/recovery for art objects on a case-by-case basis.
 - In the archival and library sectors software is required to maintain access to the data in the many archives, but also to keep copies of software products for historical purposes and inspection by the public.
 - ☞ Continue exploring the emulation and Universal Computer approaches. These facilities seem to become more common and therefore more affordable than before.
 - In the science domain, the emphasis is mainly on reproducibility of scientific output, but also on accelerating scientific discovery by re-use.

- ☞ International cooperation helps sharing the cost and finding best solutions.

GENERAL

9. From an international perspective, the Netherlands is not lagging behind in its attention for software sustainability. This is not to say that all expertise in the Netherlands is at the forefront of best practices.
 - ☞ After the initial step, continue to take further steps as required to salvage all relevant digital born heritage, particularly including software.
 - ☞ Share Dutch-born best practices and absorb international best practices in other domains.
 - ☞ Continue to monitor international developments in software sustainability.

3 Introduction

The Dutch Digital Heritage Network (NDE) and the Netherlands Coalition for Digital Preservation (NCDD) consist of partners in a number of sectors: science, archiving, libraries, art and broadcasting. In all these sectors the NDE and NCDD members are faced with many different aspects of software obsolescence, in agreement with the rapidly growing interest in software sustainability. The partners therefore quickly need more insight into this matter in order to be able to dedicate funds for a first quick inventory of the issues at stake. But the question is whether Software Sustainability is a matter of largely coinciding issues and challenges across the NCDD stakeholders' domains, or if the differences are larger than the similarities?

"What is Software?" is the title of an article by Peter Suber². It is a paper of a highly philosophical nature, written at a time one would expect the term to have been defined, but much has happened since. After twenty pages a high level of abstraction is reached while attempting to answer the question. Despite this effort, a more down-to-earth approach may be more useful: you recognise software when you see it or are confronted with it. So without knowing exactly what we are going to try to preserve, we will try to preserve it: software of all kinds in all its forms.

Why should software sustainability be an issue at all? Are communities not taking care of it already, if the software is important enough for them? Why would funders or governments have a concern for software, let alone obsolete software?

What is the nature of the challenges faced by those who try to sustain software or keep it for future reference?

The cultural heritage sector has an interest in maintaining digital born objects, artefacts, social history (why were some games or websites so attractive?) and the history of digital life (records of what happened in the past 65 years after the initial introduction of computers).

The archiving world needs to be able to provide access to archives which were based on today's obsolete technologies, hardware and software systems.

The library world hosts many digital objects that carry data, software, educational materials and so on, which need to remain accessible, readable and executable.

The gaming archives not only want to keep games playable, but want to be able to reproduce the user experience with the games. This goes beyond merely sustaining the software.

The academic world is after no less than formal reproducibility of scientific results, re-usability of code and sharing of efforts in coding, maintaining access to the information behind stored data, which requires the software with which the data were generated, or which is needed to interpret or handle the data to be maintained just as well, at least for the medium term (10-25 years).

Who is responsible for what when it comes to software maintenance or creating images for future use. Will we be using emulators³ or virtual machines?

This report is based on a research effort which was limited in scope and time. The purpose is primarily to document, by means of "use cases", the various issues regarding Software Sustainability faced by different organisations and players in the field. Some use cases have been compiled through interviews or pseudo interviews, others by completing templates. From the use cases and from activities in the international software sustainability scene, conclusions are derived and directions for further action and/or further research proposed. Despite the limitations in scope, we hope to have provided useful answers to the questions raised and practical recommendations for future work or actions.

The project also involved the organisation of a workshop, which was conducted in close cooperation with an adjacent NDE/NCDD project about Born Digital Art (focusing on CD-ROM sustainability) and placed under the umbrella of a Seminar on Preservation organised by the NCDD (June 13, 2016). During the workshop, where the main conclusions of this work were presented and discussions took place on various draft conclusions, further input was collected and merged with this document.

² Peter Suber, *Journal of Speculative Philosophy*, 2 2(1988) 89-119. Thanks to Vincent Wintermans (Unesco Persist) for pointing to this paper.

³ Emulators: software programs running on modern computers which interpret programs written for another (usually older) target machine and so cause them to be executed in much the same way as they were on the original computer hardware.

While in a parallel effort the first steps were taken by DANS and NLeSC to define a Software Seal of Approval, including criteria and formal processes, an overview of software types was produced. Although this overview was strictly speaking outside the scope of this NDE supported research, the resulting compilation of software types is appended to this document.

Matters *not* explicitly addressed in this document are:

- criteria for discarding software or keeping/maintaining software at a certain support level;
- criteria for the quality of software.

The discussion on the former criteria has not yet started, it seems. The latter criteria have been addressed in at least two documents, one⁴ by SSI, the UK Software Sustainability Institute, which also addresses maintainability and green sustainability, the other by CLARIAH-NL on requirements for software quality. That document is presently being circulated for comments.

This work is also partly based on earlier reports, in particular:

- Research Software at the Heart of Discovery - Principles of Academic Software Sustainability. 16/02/2016. Editors: Peter Doorn (DANS), Patrick Aerts (DANS & NLeSC), Scott Lusher (NLeSC);
- A Conceptual Approach to Data Stewardship and Software Sustainability by Patrick Aerts (NLeSC & DANS), Peter Doorn (DANS);
- Research Software Sustainability, Report on a Knowledge Exchange Workshop by Simon Hettrick (SSI), February 2016;
- A Unified Approach to the Preservation of Cultural Software Objects and Their Development Histories (<https://securegrants.neh.gov/publicquery/main.aspx?f=1&gn=HD-51719-13>);
- Preserving Virtual Worlds (<https://www.ideals.illinois.edu/handle/2142/17097>);
- Emulation and Virtualisation as Preservation Strategies (https://mellon.org/media/filer_public/0c/3e/0c3eee7d-4166-4ba6-a767-6b42e6a1c2a7/rosenthal-emulation-2015.pdf);
- Merritt Online Repository Link (https://merritt.cdlib.org/m/ucsc_lib_promweek);

and other documents, referenced throughout this document.

Acknowledgements

This document was only made possible by the cooperation, inputs and expert opinions of the members and staff of the NDE and NCDD, DANS, NLeSC, ePLAN, the Knowledge Exchange Group, and many more. Thanks to all of you.

In particular, I would like to thank Nina Westzaan (DANS) for her critical reading of and comments on the draft document.

⁴ <http://www.software.ac.uk/software-evaluation-guide>

4 Software sustainability in science and culture

4.1 Introduction

Software sustainability is becoming a high priority agenda item. Now that “data” have become a major topic in science, industry and the cultural sectors, people are discovering that these data either are produced through software or require software to be read, interpreted or handled. This means that software needs to be kept and maintained as long as the data are relevant. But now that software has become a matter of concern, other domains start acknowledging that software – particularly its sustainability – has been underappreciated and that some catching up is required.

In all cases the question must be asked why we should want to keep, restore and maintain a piece of software. Why retain code written for a Telefunken TR4 on Hollerith punch cards that ran for hours, when today the same action can be performed in a second using state-of-the-art software? There may actually be good reasons.

Once upon a time there was this Web page in the Netherlands, somewhere around 1992, with all Dutch websites on a geographical map of the country. Three places even had a fork projected, because they had more than one website! But the evidence is gone. Snapshots of the Internet were not made yet. From the early nineties until well into the second millennium, numerous websites have disappeared without a trace. The same holds for software. Actually, we are leaving an approximately thirty-year period behind us with less historical traces than before, because the pace of IT development got so high we could not keep up saving images of the past for future historical research.

In a recent report by DANS, “A Conceptual Approach to Data Stewardship and Software Sustainability”, three levels of stakeholders are identified with regard to these topics. The first level concerns governments, funding organisations and the like. Their interest lies in accountability and matters of general interest, notably cultural heritage. Reproducibility of scientific results and verifiability of scientific claims are most important if accountability for the expediency of allocated funds is required. Keeping an eye on cultural heritage in the digital world is certainly also something of general interest, but the role of data, digital collections and software has been overlooked in the past decades. Recovery will be a complex and labour-intensive effort.

Looking at the use cases collected in this paper, some common aspects stand out very clearly:

- Legal matters
 - (Lack of) licences
 - Copyrights and ownership
- Physical protection
- Bit rot and link rot
- (Lack of) original devices
- (Lack of) sources
- (Lack of) documentation and versioning
- (Lack of) structured processes in development
- Outdated/obsolete languages
- Outdated operating systems.

Another conclusion that may be drawn from these cases is that it makes sense to distinguish between everything that has been created (the legacy) and everything that is yet to be created: try to prevent the errors that led to the present legacy by proper education. In this way the entire problem may be brought under control.

Important differences between domains can be observed as well, one of them being the relative importance of the software code. Here a downward scale can be observed from science (where in all its basics the text of the code is the most important), via games (where the playing device is important), to born digital art (where the software, however indispensable, is only part of the issue, because what counts most is the perception of the observer).

4.2 Science

In science, it is important that outcomes involving software are reproducible. Today this is, strictly speaking, hardly the case. There are no codes of conduct, whether formal or informal, on how to refer to the software used in publications. Software versions are not commonly mentioned, if the code is referred to at all. If the software is mentioned it may still not be retrievable, ready for inspection, openly usable. There is virtually no software without errors. Some may not be essential for the scientific results (for example when the program just stops working), but other errors may actually influence the results. Still, one may reference such publications in good faith. Therefore, one needs to be able to backtrack all publications whose results indirectly depended on these references. If only for that purpose, one needs proper version control and long-term access to the software, at least as long as is required for open access to the data on which the paper's conclusions were based, i.e. about ten years after publication.

Another argument for software sustainability in science is well illustrated by the metaphor of the World Wide Web. All Web page codes have always been open for inspection ever since the first browsers emerged. This allowed for easy copying of elegant (or any) solutions to increasingly more complex Web page designs and functionalities. Today everyone knows the pace of development in this domain. In a similar manner it is likely that discovery in science will be accelerated if everyone can more or less freely use and re-use existing codes and software solutions.

Even within the context of science, the term "research software" covers many forms of software. In Computer Science, a compiler, an operating system or a scheduler may be research software. In other domains such pieces of software will be considered system software and largely fall outside the scope of the software domain. Writing code in application domains (chemistry, astronomy, physics, technical sciences, ...) is not usually seen as producing scientific output. Only the results produced with the software are acknowledged as scientific results, provided they are published in refereed papers. Currently, this is a drawback for the production of properly and organisationally well-developed software.

As mentioned earlier, the UK Software Sustainability Institute (SSI) and the Knowledge Exchange Group have explicitly started address software sustainability as a separate issue, of no lesser importance than Research Data Management; increasingly more standing organisations are getting focused on this subject, including (but not restricted to) the KNAW (Royal Academy for the Arts and Sciences), NWO, SURF, NLeSC, DANS and the NCDD in the Netherlands.

There are large research domains that should be able to take care of their own procedures and finance them in the future as well. But there are many more orphaned codes, produced by one of the many smaller four-year projects funded by universities or funding agencies. Taking care of these codes in a professional way is less trivial. Organisations such as the SSI are needed to do this job, even if it means playing the ball back to the science communities enriched with templates, guidelines, best practice based advice and more.

Applying software sustainability in science involves a few steps. Just to give an impression, without an attempt to be exhaustive:

1. Determine if the software is worth keeping and/or maintaining, and if it is:
2. Determine to what extend the software is to be maintained;
 - a. Just keep the (text of the) code and/or
 - b. Keep a binary
 - c. Maintain accessibility
 - d. Maintain usability
 - e. Continue updating or more;
3. If a code has been used in a publication it must be maintained to a level that renders it usable on currently available computer equipment for at least ten years;
4. Maintained software must be citable, findable and accessible.

4.3 Libraries

Libraries are confronted with a transformation from collecting and making findable and accessible hardcopy, mostly paper, documents (books and other printed materials) to collecting and making

available digital objects of historic potential. Organisations and even countries have different policies as to which digital objects qualify to be kept by libraries and which objects are the concern of other organisations. In the Netherlands, the National Library keeps digital copies of publications and books but also large amounts of CD-ROMs. Games, however, are kept by Beeld&Geluid, a unique organisation. In the US, games collections are collected by museums rather than libraries.

Since the early 1990s, digital documents have entered library collections, but in a large variety of formats, many of which are obsolete today. The software to read these documents is just as obsolete and will not perform or run at all on today's computers. This poses a threat to the readability of these many documents. In addition, libraries may want to keep software as part of the history of computer programs, as examples of an era and the way we did things then. But that function only works if the software actually runs, on either emulated or virtualised machines.

In the Dutch case, where the National Library also is in fact responsible for CD-ROM collections, additional issues arise. Working CD-ROM players are decreasing in number, the CD-ROMs will fail to be read due to storage errors (bit rot), and so on. Their concern is therefore how to make copies of CD-ROMs in a sustainable format, and how to subsequently run the software. Again: emulation, virtual machines, etc.

The Dutch National Library was one of the early adopters of emulation as the leading means of making software sustainable. It demonstrated that emulation based on the Universal Virtual machine (UVC) concept is a viable option for long-term digital preservation.⁵ The research conducted, however, focused on methods to preserve document types of data, which does not require full emulation. Preserving software using this method does require full emulation of older computers and runtime environments.

As emulation used to be an expensive way of running outdated software, it was not obvious at the time that this methodology would be sustainable. Today, however, the interest in older computer systems, their operating systems, architecture and specific characteristics has got the attention of many people and freeware emulators⁶ are now provided for many types of computers, including gaming systems.

4.4 Archives

In a recent article in the Dutch daily newspaper *De Telegraaf*⁷, reporting on an earlier article by *Binnenlands Bestuur* (a publication for professionals in public government), it was reported that a large number of city administrations suffer from all sorts of data loss due to ill-kept information records. It concerns inaccessibility due to obsolete formats, programs no longer working, bit rot and more. The article is an apparent reflection of the general ignorance concerning long term data stewardship and software sustainability, but also – hopefully – the start of government seeing the light: policies should be designed and put in place for proper long term data management and software sustainability.

The issues at hand in the archiving world do not seem, however, very different from those already mentioned for libraries, although it should be noted that some archives may have been constructed using complex and proprietary data and record formats, which are not easily reconstructed without the original software, at least for the sake of conversion.

4.5 Broadcasting, Sound&Vision

Beeld&Geluid (B&G) (Sound&Vision) is part of the NCDD consortium and it has a specific task regarding games preservation in the Netherlands. B&G has its specific issues regarding software for copying analogue data sources, such as sound, film, movies, photographs, and texts on

⁵Lorie, R. A., 2002. The UVC: a Method for Preserving Digital Documents – Proof of Concept. IBM/KB Long-term Preservation Study. Amsterdam: IBM Netherlands. Citation after Wikipedia's lemma on Universal Virtual Machines (https://en.wikipedia.org/wiki/UVC-based_preservation). More references can be found there.

⁶ See for example <http://www.emulator-zone.com/>

⁷ De Telegraaf on line 20-05-2016.

http://www.telegraaf.nl/binenland/25829077/___Gemeenten_zitten_met_floppy_s___cd-roms_in_hun_maag_.html

carriers such as LP's and smaller records (back up to glass records), many types of tapes containing unique and original audio-visual materials, most of which are used in radio or television broadcasting and today of course also on the Internet. It appears that the majority of software is made by third parties, although some software is homegrown. The original materials have either already been copied to modern magnetic carriers in sustainable formats or are on the verge of being processed in this way. In that sense software sustainability is less of an issue for B&G. One obvious exception is the role that B&G plays in games preservation. Due to the nature of games (much visualisation and sound) it has been decided among B&G and the National Library that B&G will take care of the significant collection of games of Dutch national origin. To these activities all issues apply that apply to gaming software in general, which is discussed in the next paragraph.

4.6 Game preservation

Gaming is the first example of digital objects where the role of the software is essential but only part of the total play. Games have been created on and for about all computer systems ever designed, if only for the engineers to have some fun during testing and other work. However, in gaming, the very device on which the game runs is an essential part of the experience. Hence having the software available is only part of what needs to be achieved. The software itself or its maintenance for the future has hardly any value without the devices that created the experience. You can play a Nintendo game on a laptop, but for what purpose?

In industry, compatibility with earlier versions of software may be important. But in gaming the experience of the game is just as important as the code on which it is based. The very devices, from large machines in gaming halls, to different kinds of handhelds and TV screens are all relevant if future generations want an impression of what the game entailed and what drove people into buying and playing those games. And interactive Internet games complicate matters even more.

The use case for gaming is extra extensive because of the implicit complexity of the interaction of the software with the device it runs on and the environment that takes part in the game, such as in community games played over the Internet.

Gaming software is also extraordinary in the complexity of the citing strings required to know which game is actually referred to. Addressing games requires the game software version, the make version, the device on which it runs, the format and the version of the cartridge on which the game was stored and put into the device, etc. Yet there are even more complex software environments: they can be found in the world of art.

4.7 Museums: Born digital art

Works of art created on or by computers go back a long way. As early as 1949(!)⁸ on the Australian CSIRAC computer a programmed tune was played. Because the machine was restored 50 years later, that tune can still be played in its original form. Most likely other programs have been written to use computers for other purposes than pure, applied math: computer art, art which was born digital. Computers were not yet transistorised and used vacuum tubes until the late fifties, so the hardware remained partly analogue at the time, but the logic was digital. Art and computers went hand in hand, although for a long time real artistic expressions on computers remained in the background, more as a gimmick than as a serious attempt to create artworks in a new manner.

What adds to the complexity of digital art is the fact that artists tend to express their creativity by doing things differently. But although most artists create artefacts for an indefinite future, when it comes to digital born art, which depends on languages, operating systems and computer hardware that becomes obsolete with every innovation cycle (typically three years), the artefact becomes inaccessible/unusable rather soon. Moreover, in contrast with the scientific software domain, where having access to the written code at least would be enough to understand what the author wanted to express or calculate, the perception of the viewer is the most important aspect of an artefact. This means that having the software that forms one of the basic elements of a born digital artefact, is not enough to maintain the object and use it as intended.

⁸ <https://www.youtube.com/watch?v=yxDQSluWaMs>
<https://www.youtube.com/watch?v=6F44b60Mq30>

With a view to recovering digital born art objects, LIMA has extracted a specific workflow. Yet it seems difficult to set general rules and processes for born digital art sustainability. Most likely the decision to spend funds and efforts on sustaining a particular piece of art will have to be taken on a case by case basis (comparable perhaps to museums' processes for deciding whether or not to buy a certain piece for their collection). The cost of recovering or rewriting software to keep an artefact alive may be very significant.

4.8 Digital history and archaeology

A specific domain is that of digital history, or the history of digital computing⁹. Now that the age of computer development extends well past its diamond anniversary, the interest of historians in this field is growing. At the same time, it is found that researching this is like looking into pre-history: just as history starts with the written word, of which traces can be found in stone, clay tablets, on parchment and pig skin, and pre-history is defined as the period before any written remains, so does the history of computers and digital data start out in pre-historic times with written (and physically readable) objects lacking!

The *Digitale Stad* (Digital City) project is an outstanding example of what it takes to restore lost information and, most importantly, lost experiences. At the time, *Digitale Stad* was a very advanced and progressive endeavour, of which some remains are still around, but whose basis has largely disappeared. It may, however, be considered an extended museological object because it prototyped an era of early development for the Internet, communications and social forums.

From an educational point of view, the history of digital computing is an important new domain of scientific research, comparable to the history of mathematics. But as was mentioned before, many of the historical objects no longer exist. The interest, however, is so great that significant efforts are being made to restore e.g. entire obsolete computer systems.

4.9 Common issues across all domains

Common issues encountered across all domains are:

- Legal matters
- Bit rot and link rot
- Lack of original devices
- Lack of sources
- Lack of documentation and versioning
- Old/obsolete languages
- Old operating systems
- Lack of licences
- Copy protection
- Copy disallowed

4.9.1 Legal matters

The most common problems experienced in maintaining, curating and/or running older software are legal matters. This is exemplified by the case of the CD-ROMs at the Dutch National Library. The National Library has the legal task and obligation to preserve documents, publications, books, and similar artefacts, including computer programs. In order to counteract bit rot on CD-ROMs and the increasing obsolescence of reading equipment, they obtain so-called images of the contents of the disks. Here, the first violations may occur: the software may be copy-protected and copying the contents to another medium may be forbidden by copy protection law. Once these hurdles have been passed the image is to be run on a virtual system running some obsolete version of an operating system (which may be obsolete as well), say Windows 3.11 or Windows 95. Without a proper license this is illegal too.

⁹ In the context of this document, this includes computing on pre-transistor type computers.

These problems are common to all attempts to run old software on old systems and is an issue discussed at an international level.

The organisation UNESCO PERSIST is presently concerned with working out deals with global software vendors for the legal use of their legacy software. This should alleviate the issues concerning the legal use of obsolete versions of operating systems, such as Windows 95 or old versions of Mac OS, as well as the use of old versions of Word, MacWrite, Excel and the like. Other parties¹⁰ are considering the creation of license banks, where old software licenses not in use could be re-used.

A term that appears to be used in this context is "abandonware"¹¹.

The Wikipedia article on abandonware distinguishes the following types, which are – slightly shortened – are repeated here because it is worthwhile reading:

- Commercial software unsupported but still owned by a viable company
 - There are abandonware websites where copies of unsupported and discontinued operating systems made by Microsoft and Apple are archived and made available, as well as rare development builds of such operating systems that have been leaked by the media or technicians working for said companies. After Windows XP was discontinued in April 2014, numerous websites started providing the operating system for free, despite the fact that its usage and popularity prevent it from being considered abandoned.
- Commercial software owned by a company no longer in business
 - When no owning entity of a piece of software exists, all activities (support, distribution, IP activities etc.) in relationship to this software have ceased. If the rights to a software are non-recoverable in legal limbo ("orphaned work"), the software's rights cannot be bought by another company, copyrights cannot be enforced, etc. An example is Digital Research's original PL/I compiler for DOS, which for many years was considered to be without an owner (now probably owned by Novell).
- Shareware whose author still makes it available
 - Finding historical versions, however, can be difficult since most shareware archives remove past versions with the release of new versions. Authors may or may not make older releases available. Some websites collect and offer for download old versions of shareware, freeware, and (in some cases) commercial applications. In some cases, these sites had to remove past versions of software, particularly if the company producing that software still maintains it, or if later software releases introduce digital rights management, whereby old versions could be viewed as DRM circumvention.
- Unsupported or unmaintained shareware
- Open source and freeware programs that have been abandoned

Obviously, the legal aspects of these types of unsupported software may vary hugely. But the important part is the recognition that new legislation may be in order to cope with this legacy.

This is also true for yet another domain of archiving: Web archiving. Somewhere in between archiving data and sustaining software lies the domain of the Internet (mostly the World Wide Web). Only a very limited number of Internet archives are in existence today. The Dutch National Library has one.

However, legal restrictions prevent it from making this archive open to the general public: only on the premises of the National Library in The Hague can one browse this wealth. This situation needs to be mended.

4.9.2 Bit rot and link rot

"Bit rot" refers to the deterioration of the magnetic medium or carrier of data ("the bits falling down").

"Link rot" refers to Internet links used on the World Wide Web that are no longer active/existent. If anything in the discussion of preservation should sparkle a sense of urgency, it is bit rot. But link rot is equally frustrating for Web heritage.

Even the term "software rot" seems to be in use in some communities¹². In this case the rot is largely virtual, because nothing really disappears. It is the environment that changes without the particular

¹⁰ NLnet, private communication.

¹¹ <https://en.wikipedia.org/wiki/Abandonware>

¹² https://en.wikipedia.org/wiki/Software_rot

piece of software changing with it or changing at the same pace. Software becomes, or may be just perceived as, slow. To add to the confusion, sometimes the term bit rot is used to this phenomenon of software rot¹³.

Instead of bit rot, the term "data degradation" is sometimes used. Obviously, data degradation is often the result of physical changes in storage media. And those changes are referred to as bit rot. The causes of bit rot are different for different media. In tapes, the magnetic orientation of stored bits may change, in other media it may be the physical medium used to contain the magnetic properties that may deteriorate. Unreadable magnetic media can only be restored, if at all, at the expense of significant investment cost.

Generally, bit rot is prevented by timely copying materials to other media or recopying them to the same storage medium (and repeating this procedure regularly). But this requires access to the magnetic source in the first place (sometimes complicated due to copy-protect measures).

Link rot occurs very frequently. Whole websites may disappear (if only because the domain fee has not been paid) or change, for example because a new CMS is introduced. This complicates Web content conservation. There are practical ways to avoid link rot, for example by using DOIs (Digital Object Identifiers) and PURLs (Persistent Uniform Resource Locators), but those are mostly used by parties who *a priori* want to share information indefinitely. They cannot be used automatically; one has to invest in getting links persistent. For something like historical curiosity (what did the Web look like in the 1990s), these methods do not solve anything.

In a recent article in the Dutch newspaper NRC,¹⁴ Herbert van de Sompel¹⁵ (Los Alamos) and Frank Huysmans¹⁶ (UvA) were interviewed on the topic of digital amnesia. After only one year, 10-15% of the links in scientific publications are broken, they claimed. If one were to visit the website of a former prestigious department store in the Netherlands (V&D) today, after it has gone bankrupt, nothing has remained of it. With that, much of the historical knowledge about this company's operations, successes and failures has gone too.

So link rot as such is difficult to avoid when doing general research on the Web. Internet Archives have come to the rescue and help a bit. Web files are stored in there in snapshot style, but link rot hampers proper functionality of the linking websites significantly. Sometimes matters can be restored, if the link has changed address but still exists, or if the material can be recovered from other sources.

4.9.3 Lack of original devices

Tapes, floppy disks, hard disks, optical disks, cards (paper and magnetic) require original peripheral equipment to expose their data. If only for making a one-time copy. The peripheral equipment in turn often requires drivers on computers, running under specific (versions of) operating systems to function. Old computer systems are a necessity, both for reading data (and software) and for executing obsolete software on appropriate systems.

Worldwide, there are a few computer museums with working systems. Old and very old computers have been restored or even rebuilt, often just for demonstration purposes, but they can be used for practical purposes too.

When it comes to more "modern" systems, such as original Personal Computers, Commodores, Ataris, IBM PCs, Apple Macintoshes, an entire industry has emerged refurbishing such systems, both the hardware and the cases, replacing unrepairable broken parts with similar ones from other systems or with newer parts with the same functionality.

Eighties and nineties mainframes, mini-supers and supercomputers are definitely rarely found in museums. They are heavy and energy consuming.

In the Netherlands there is an organisation dedicated to computer heritage, SCEN (*Stichting Computer Erfgoed Nederland*¹⁷), whose members represent most of the public and private computer collections in the country. Individual computer collections can be found at the Netherlands Institute for Games and Computers, formed from the *Bonami Spelcomputer Museum* and The Computer Museum. They now share a common exposition space in Zwolle¹⁸. However, there are more museum-quality collections. There is the Apple Museum¹⁹, the *Tehuis voor Bejaarde Computers*²⁰, a private collection with surprising

¹³ The Jargon File: https://en.wikipedia.org/wiki/Jargon_File

¹⁴ <http://www.nrc.nl/handelsblad/2016/05/31/vechten-tegen-digitaal-geheugenverlies-1623567>

¹⁵ public.lanl.gov/herbertv/ and https://en.wikipedia.org/wiki/Herbert_Van_de_Sompel

¹⁶ www.uva.nl/over-de-uva/organisatie/medewerkers/...huysmans/f.j.m.huysmans.html

¹⁷ <http://dutch-computer-heritage.com/>

¹⁸ <http://www.gamesencomputers.nl/>

¹⁹ <http://www.applemuseum-nederland.nl/>

elements (PDP-11 computers from 1970-1985, DEC's Vax, Gould PowerNode, workstations and servers from SUN, DEC, IBM, Tectronix, and more). The University of Amsterdam Computer Museum²¹ has a large variety of mainly smaller devices and peripherals and a few minicomputers, such as a PDP-8 (1965). ING Bank had a computer museum in their Arnhem office, but since that site was closed it is unknown what happened to the legacy, which included an IBM 1401. Other collections are Erik Klooster's *Computermuseum*²² (a private collection of mostly PC type systems), and the *Vitrinemuseum*²³ (a small collection from TU Delft's EWI Faculty).

In fact, there turns out to be an inventory of computer systems hosted by SCEN²⁴.

Many of the objects in these collections have no actual function, because they represent only parts of a total, but a few objects, mostly in the PC domain, still work. It would be useful to have a list of working systems and peripherals available, in case data or software considered relevant needs to be rescued. One could even consider setting this up on a European scale, because the UK, France and Germany do have more extensive collections of working systems including larger-size systems (like mainframes).

²⁰ <http://www.bejaardecomputers.nl/>

²¹ <https://ub.fnwi.uva.nl/computermuseum/>

²² <http://computermuseum.50megs.com/>

²³ <http://www.vitrinemuseum.ewi.tudelft.nl/>

²⁴ <http://dutch-computer-heritage.com/COLLECTIONS.htm>

5 Need for a software sustainability facility

5.1 Introduction

One of the questions raised by the NCDD was how to organise software sustainability in the Netherlands. Is there a need for some sort of formal software sustainability organisation? A few observations should give enough direction to take action.

5.2 Support from the science community

In a survey conducted in the Netherlands among over 10,000 researchers in 2015, the question was asked whether the concepts of Data Stewardship, Research Data Management and Software Sustainability were familiar to them and if so, what they thought should be done about them. A vast majority was not acquainted (let alone involved) with any of these terms, but over 80%(!) of the respondents considered the topic important enough to warrant a national or international policy. In other words, the science community at least supports that something concrete should be done about software sustainability, both in the interest of the general public and theirs.

5.3 Past versus future

In order to tackle the problem of software sustainability, two discussions are needed: one concerned with the question what to do with the vast amounts of legacy software and software still used today, the other about new legacy to be created by taking preventive measures. So far all of the discussion has concentrated on the legacy: what to keep, what to let go, what to actively maintain, what to keep as is, how to revive older software, how to make it accessible, and so on. This discussion has to continue until the appropriate measures have been taken and the risk of losing important tools and products is minimised. But can we prevent further growth of the stack of legacy software?

This is where education enters the discussion, aimed at today's software developers as well as students.

The most important way of contributing to the solution of software sustainability issues is to teach students, before teaching them any programming languages or other programming tools, how to write code that will be intrinsically maintainable at minimum cost. Matters such as version control, documentation (not just user manuals) and modular coding should be stressed as being an integral part of good coding practices. Of course, this should also be taught to people who do their coding today. It may strongly contribute to mitigating future sustainability issues.

5.4 Expertise in an international context

Maintaining or recovering software for renewed use requires considerable expertise. So does formulating advice to maintain software yourself or in your group or with your peers. A focal point where this type of knowledge and expertise is available, or which actively promotes best practices, would surely be very helpful. One of the centres with the most experience in sustaining software is the Software Sustainability Institute in the UK. Although its scope is limited to the scientific domain, much of what it is doing is broadly applicable and readily available.

At the Open Science Conference in April 2016, DANS presented the suggestion to have all European countries define national Software Sustainability Initiatives and to create an overarching European Software Sustainability Infrastructure on top of that. The form or extent of each of those initiatives may vary from light (just a contact point) to heavy (an institute), but the European infrastructure will connect them into one knowledge infrastructure focused on this very topic. ePLAN (in the Netherlands) and PLAN-E (in Europe) have adopted this idea and will help promote it and perhaps also take the necessary steps towards implementation.

6 Use cases

6.1 Introduction

At the heart of this report are the use cases selected from different domains so as to cover as many of the practical issues software preservation organisations are confronted with as possible. Use cases have been gathered from various people and different sources. The cases have been condensed to 1 or 2 pages wherever possible, using a single template for all of them. This should enhance the readability and accessibility of this document.

Use cases help define the scope of the issue – “software sustainability across science and heritage domains” - and should promote understanding of the matters at stake. Although they are examples and snapshots and not all issues will be covered by them, together they reflect the broadness of topics at hand and the nature of the matter, at least from the perspective of “users in the field.” There may or may not be links between different use cases, as they have been collected from a range of sources.

6.2 General conclusions based on the use cases

There are large variations in the nature of software products that play a role in the sustainability discussions in the public cultural sector. There will definitely not be a one-size-fits-all solution to sustainability issues. But from the various use cases brought together here, four useful conclusions can be drawn:

1. There are serious legal problems to be overcome before software from the past can be lawfully used;
2. Software maintenance goes well beyond keeping a copy of the text;
3. Having software properly maintained does not guarantee its sound use;
4. Criteria for keeping or discarding software are lacking.

These conclusions will be addressed in the discussions and the summary of this document.

6.3 Use case: Pineapple, a semantic search engine for historians

Author: Dirk Roorda (DANS)

Discussed with: Alexander O’Connor (<http://oconnoat.com>)

Time frame: 2012-2016

Abstract: The European FP7 project CENDARI (<http://www.cendari.eu>) has brought together a number of heterogeneous historical archives. There were two focuses: (1) medieval manuscripts and (2) World War I. What the historians needed was a search engine for concepts occurring in archived materials. The software, consisting of a UI (Pineapple) and CENDARI, a semantic repository, has been developed in the course of the four-year FP7 project. The issue at hand is what to do to maintain this code.

-
- ✓ The software:
The software consists of PINEAPPLE (the UI) and CENDARI, a semantic search engine, structured around collections of documents, in which entities with semantic relations and attributes are represented. Its purpose is to disclose – mainly to historians – a heterogeneous collection of documents and archives by finding concepts rather than strings in full texts. It was developed in a four-year FP7 project with many partners. The net development cost was 2 FTE.
 - ✓ Domain software specifics:
Different groups use different programming languages, either due to local rules or due to the different skills of the international participants. In practice both Java and Python were used, glued together through API’s.
There was a common OS: Linux (Ubuntu).
 - ✓ Software dependencies:
Semantic technology is dependent on “triplestores”. Other dependencies mentioned are Virtuoso, Prolog and CKAN.

- ✓ Software structure:
Cendari and Pineapple consist of a collection of loosely coupled services. The backend works with a unifying API. The frontend builds services on top of that, hiding the specifics of the backend. This functionality wrapping results in an ecology of services.
- ✓ Development environment:
The system was set up using Github (with CENDARI as institutional owner). The final documentation is also available there. The source code docs, deployment recipes and user manuals were added at a later stage of the project.
- ✓ Software distribution:
The software is not distributed. There is a single installation that serves the community. The user interface is a Web application.
- ✓ Re-use and development:
Basically, the existing triplestore software was used. The software itself was designed from scratch.
- ✓ Organisation:
Many parties were involved, from three different professional backgrounds (archivists, computer scientists and historians). It took one year to get a time frame for the development and to collect requirements. An agile-ish method of iterations was used to compile it. Because there was a lot of back-end programming required, the developers could not rely on purely user-centric design methods. One complication was the big turnover of people without a stringent documentation policy in place. Also, the collections at hand were quite heterogeneous, with different metadata schemes.
- ✓ Users:
Requirements and pseudo-testing were realised by using scholars as proxy users. Maintaining provenance and ingesting the materials into the repository, including all conversions, was a major requirement. Users started appreciating the product only after initial performance frustrations.
- ✓ Sustainability aspects:
Concerning re-use, the software is intended to be used indefinitely, with modifications as needed. The software is presently maintained by the Cendari Sustain Working Group, which is part of DARIAH. Prior to the start of the project a Data Sharing Agreement and a Licence were signed by Cendari and DARIAH-EU to secure future maintenance. The knowledge-intensive software system should continue to follow the needs of the community, which means that maintenance involves more than keeping the lights on. The software is owned by the Cendari Working Group. It is unclear how ownership will evolve after the Working Group will be dissolved. Ideally, a "CENDARI in the box" (virtual machine solution) would have come into existence, but this has turned out to be pretty complex. Because the machine would also have to be patched with security updates, there still is no guarantee that the software will continue to work. The community, however, seems more concerned with keeping the data than with keeping the software.
- ✓ Referencing:
The computer scientists have published about the data model and the wrapper API:
<https://www.researchgate.net/publication/281835833> *The Taste of Data Soup and the Creation of a Pipeline for Transnational Historical Research*.
For historians it seems too early for the software to be quoted. No special measures have been taken to reference the software beyond its residence in Github.

6.4 Use case: CATCH (Continuous Access To Cultural Heritage)

Author: Patrick Aerts (DANS)

Discussed with: Jaap van den Herik (<https://www.universiteitleiden.nl/en/staffmembers/jaap-van-den-herik>)

Time frame: 2004-2016

Abstract: The aim of the CATCH programme is to make the collections of museums, archives and historical associations and the like more accessible and help heritage managers to do their work more efficiently. Software development was an essential part of the project. This use case describes the way the software came into development and the measures taken for future maintenance.

- ✓ The programme:

The CATCH programme was carried out between 2004 and 2016. It was a brand new concept to rejuvenate methods of access to and disclosure of cultural heritage in general and serve researchers, scholars, "service organisations" (i.e. the staff of museums, archives, data and music banks, etc.) and the general public in particular. Under the programme some 18 projects were granted. Each project was carried out by at least one post-doc, one PhD student, a programmer and a principal investigator (the applicant). Also, at least two parties had to be involved: one from the science/research domain and one from the cultural sector (private or public). At the beginning of the programme a zero measurement was made to outline the current situation in order to enable comparisons at a later stage. Basically, that measurement returned a zero-facilities situation: some parties (in 2004) did not even have Internet access.
- ✓ The software:

Within each project software was developed to archive, disclose, list and make otherwise accessible a special case of cultural heritage objects. The software was to serve mostly scholars, staff and the general public. There were large variations in the types of software to be developed, from a song base to audio services for tourists in museums.
- ✓ Domain software specifics:

As described above, the software had to serve different communities. Analysis tools for scholars had to be implemented as well as scaling for access by the general public.
- ✓ Software dependencies:

The most critical dependencies concern the operating systems and the browser versions.
- ✓ Development environment:

Originally, when the programme started, most of the software consisted of standalone programs. Most of them ran under Windows. Later on, more and more Web based code was developed. No stringent git type procedures or version controls were used across the programme, but versioning was internally used for communication and progress reporting. Documentation was kept in a separate database. Because the software had to be used by the general public (including tourists), the codes had to be proven full/fool proof.
- ✓ Software distribution:

The software is not distributed. There is a limited set of instalments that serve the three user groups.
- ✓ Re-use and development:

Standard Web tools were re-used, but most of the specifics were developed from scratch.
- ✓ Organisation:

A lead programmer was appointed, who managed all project programmers for the sake of uniformity, interoperability and programming strategies. The post-docs and PhD students involved had to spend 60% of their time on site at the relevant cultural sector party. This was to ensure they would perfectly understand the culture, language and methodologies of the organisation while designing and implementing the software.
- ✓ Users:

For testing purposes, early adaptor scholars and staff were involved rather than the general public. In the course of the project the codes could be adapted to new insights and experiences. In particular, the students had to learn from the software: for example, to learn to group painters in categories and/or find interrelations between painters, personally or technologically.
- ✓ Sustainability aspects:

Concerning re-use, again, the software was intended to be used indefinitely. Regarding future sustainability, it is to be noted that in addition to the programme funds additional financing was received for valorisation purposes (valorisation being the term used for the deployment in practice, beyond the scope of the project). This allowed for a stronger involvement of the end-users of the software, i.e. the service providers (the cultural sector parties involved). As a consequence, they are now basically made responsible for future maintenance and thus for the sustainability of the software. If this is enough guarantee is not *a priori* clear, but to a large extent it is obviously in their own interest that they will take this task seriously. In addition, a software broker was involved for some time, but this seems not to have made much difference in this respect.

Regarding the coding aspects of the software, the code was written in a state-of-the-art way due to the involvement of professional programmers. Beyond that – and the documentation policy – no further precautions were taken to guarantee ease of maintenance.

✓ Referencing:

All projects involved a PhD scholar and all finalised their research with a thesis. So the minimum referencing possibility to the software could point to that thesis. But frequently separate papers were produced along the way, which also allow for formal referencing. As a rule, no git type archives have been used for direct referencing, although in an individual case this may have been the case.

6.5 Use case: Software for access to legacy CD-ROM collection

Author: Johan van der Knijff (National Library of the Netherlands, KB)
Time frame: 2012-2016

Abstract: The National Library of the Netherlands (KB) owns an extensive (> 15,000 items) collection of offline optical media carriers. The majority (about 63%) of these carriers are CD-ROMs; the remainder is mostly made up of audio CDs (32%) and DVDs (4%). In an effort to preserve its extensive collection of offline optical media, the KB is investigating the use of emulation as a means of providing long-term access to legacy CD-ROMs and DVD-ROMs. Since many CD-ROMs in the collection rely on technical environments that are now obsolete, the availability of the original installers of old operating systems, hardware drivers and legacy software is an essential prerequisite of an emulation strategy.

✓ **Problem description:**

The contents of these carriers are standalone publications, or supplementary to printed books. The long-term accessibility of this collection is threatened by three factors:

1. Data loss due to gradual degradation of the physical carrier.
2. Gradual obsolescence of the hardware to read the physical carriers. In recent years the use of CD and DVD drives in computers has become less common, which means that over time it will become increasingly difficult to obtain the hardware that is needed to read these carriers.
3. Most of the CD-ROMs contain legacy software that requires a specific technical environment (hardware and operating system). This becomes problematic if the original environment is no longer available, in which case the content may not be rendered or interpreted in a meaningful way.

✓ **Required steps:**

Action is needed to keep this collection accessible for future generations. More specifically, the following steps are necessary:

- Stabilise the information on the physical carriers.
- Migrate the information to a modern carrier.
- Re-create the technical environment that is required to render the content.

✓ **Proposed solution**

For CD-ROMs and DVDs, the stabilisation and migration steps can be realised by creating (ISO 9660/UDF) disc images of the physical carriers²⁵. These images can then be stored to hard disc. For CD-ROMs and DVD-ROMs, emulation is the most practical strategy to re-create the required technical environments²⁶. Using emulation, old software can be run on a virtual machine that runs inside a modern environment (e.g. Windows 10 or Ubuntu). This approach is now widely used to provide access to legacy software. In the KB's case, the aim would be to give users in our reading rooms access to the CD-ROM images, with the ability to load an image in one or more pre-created emulated environments.

✓ **Software engineering:**

Based on a limited sample, it appears that most of the CD-ROMs in the KB's collection have dependencies on old versions of Windows or MS-DOS operating systems. Common environments include MS-DOS 5.0, Windows 3.1, Windows 95, Windows 98, Windows ME and Windows NT. Although the KB has never systematically collected old software, installation CDs of most of the above operating systems from Windows 95 onward could still be located in our IT department's software archive. The older operating systems were not as readily available, but installers can

²⁵ Audio CDs must be handled differently: unlike CD-ROMs and DVDs, the data on an audio CD isn't formatted according to a real file system. As a result, the audio stream must be "ripped" to audio files (e.g. in WAV format).

²⁶ Images of audio CDs and video DVDs can be rendered with standard media player software (e.g. VLC Media Player), without any need for emulation.

often be bought second-hand on websites such as eBay. Often, these older operating systems have additional dependencies. Some examples:

- MS-DOS and early Windows versions required a third-party driver for accessing a CD-ROM drive. Even within an emulator this driver is necessary, because without it the operating system will not recognise the file system on (an ISO image of) a CD-ROM. These drivers can be difficult to obtain legally, although downloads can be found on some computer hobbyist websites.
- In some emulators, older Windows versions require third-party display drivers in order to get sufficient graphical resolution. These drivers can be difficult to track down as well.
- The original installation CD-ROMs of Windows 95 and Windows 98 are not bootable, which means that a separate "Start-up Disc" (a 3.5" floppy disc) is needed to install these operating systems.

Moreover, the content of some CD-ROMs requires additional legacy software, such as (old versions of) Microsoft Office. Since the above dependencies are mostly stored on vulnerable physical data carriers (CD-ROMs, 3.5" floppy disks), these need to be imaged and stored at a safe, sustainable location as well.

✓ Current status:

The work on the preservation of optical media at the KB is currently in an early research phase. So far we have created ISO images from a small sample of carriers. We created a limited number of legacy environments (MS-DOS 6.0, Windows 3.11, Windows 95 and Windows 2000) as virtual machines within Oracle VirtualBox. This enabled us to successfully render the ISO images. Although the VirtualBox setup provides a useful proof-of-concept, it is not ideally suited for providing access in a reading room setting. A better solution would be the "Emulation as a Service" framework by the University of Freiburg, and we will investigate this more in-depth at a later stage. Finally, we have made an initial effort at preserving the software that is needed to render the CD-ROMs by making disc images of the installation CD-ROMs and floppy disks.

6.6 Use case: NDL (Naive Discriminative Learning), an implementation in R

Author: Dirk Roorda

Place, Date of interview: Tübingen, 23-2-2016

Discussed with: Harald Baayen

(<http://www.sfs.uni-tuebingen.de/~hbaayen/contact.html>)

Abstract

Naive Discriminative Learning implements classification models based on the Rescorla-Wagner equations and the associated equilibrium equations. This package provides three kinds of functionality: (1) discriminative learning based directly on the Rescorla-Wagner equations, (2) a function implementing the naive discriminative reader, and a model for silent (single-word) reading, and (3) a classifier based on the equilibrium equations. The classifier is provided to allow for comparisons between machine learning (svm, TiMBL, glm, random forests, etc.) and discrimination learning.

✓ Problem description

The software concerns the modelling of learning behaviour according to basic principles of error-driven learning that characterises animal and human learning. The learning agent is eventually modelled as a network with weighted edges, and behaviour is modelled using choice paths through the network while being stimulated by activation inputs. The network may have thousands or millions of nodes, so modelling requires computing power. One of the interests of the project is to reproduce language production and language comprehension by humans in such a way that actual behaviour of humans while learning can be reproduced by the model, including typical mistakes, the amount of effort needed to learn a specific pattern, and so on. The project is the Alexander von Humboldt Professorship granted to Harald Baayen (€5 M for 5 years).

✓ Software engineering:

The program is written from scratch in R and C++ and operates under Linux, although NDL is basically cross-platform and being continued as a newer version, NDL2, which is in development and currently operates under Linux only. No strong version control system has been implemented. The program uses rccp (seamless R and C++ integration) see: <https://cran.r-project.org/web/packages/Rcpp/index.html>.

Because it is an R package written in C, it is one module of software.

Faithful interpretation of a new kind of classifier and performance were the most important implementation criteria.

✓ Software distribution:

CRAN (R's package manager) was used for distribution. More info can be found on the NDL home page: <http://cran.xl-mirror.nl/web/packages/ndl/>. The software has a GPL-3 license and is Open Source (the source is contained in the R package).

✓ Organisation:

A small group of people has been responsible for the creation and development of the software for several years. The initial concept was created by R. Harald Baayen with contributions from Petar Milin and Peter Hendrix. After that, the algorithms and packaging of the software in R were done by the same people, plus Cyrus Shaoul, Antti Arppe. All software requirements are driven by the research to be conducted and the needs of packaging and distribution.

✓ Referencing:

The software makes it possible to replicate the simulation results for Experiment 1 of [Corpus linguistics and naive discriminative learning Baayen et al. \(2011\) DOI 10.1590/S1984-63982011000200003](#) and is citable by proxy of the original paper that describes the model. A reference of this paper is on the README file of the R package, including a download link for it: Baayen, R. H., Milin, P., Filipovic Durdevic, D., Hendrix, P. and Marelli, M. (2011), An amorphous model for morphological processing in visual comprehension based on naive discriminative learning. *Psychological Review* 118, 438-482. Download from here. (accessed 2016-02-24). Furthermore, there are also github repositories for using NDL on specific datasets, such as <https://github.com/cyrus/ndl-examples>.

✓ Users:

The developers/creators of the software are also the principal users (software by researchers for those same researchers). The rest of the audience consists mainly of linguists (computational and corpus linguists, psycholinguists). The spectrum of users is most likely pretty limited: most users

quickly proceed beyond the advanced stage. Conferences, videos (<https://www.youtube.com/watch?v=ee3p4canV8k>), scientific articles and collaboration with Harald Baayen and his group are the most important vehicles to attract new users.

✓ Documentation:

The documentation can be found inside the R package, readily available through the R interface and also by browsing the source package. The documentation mainly consists of research papers: no tutorials, manuals or cookbooks. A few things regarding installation can be found on stack overflow.

✓ Sustainability:

The software is intended for (re-)use and for the long term; it is presently maintained by a member of the development group. No further precautions presently apply.

However, today there is a stable NDL package with further active development of the NDL2 package. This is not yet ready for CRAN distribution. NDL2 will be overhauled by professional software engineers before release. The software is owned by Harald Baayen and his team, and those who have contributed in the past and who are still involved with it formally (inside or outside the Harald team).

Packages on CRAN meet quite a few requirements that are beneficial for sustainability, one of them being cross-platform properties.

6.7 Use case: eSTeP (eScience Technology Platform)

Author: Patrick Aerts

Place, Date of interview: Amsterdam, 17-3-2016

Discussed with: Mateusz Kuzac

Abstract

eSTeP, the eScience Technology Platform, is the development environment for the software created and developed and/or certified for further use by the Netherlands eScience Center (NLeSC). For NLeSC, eSTeP should be the exemplary way software is to be created, maintained and distributed for direct and future use and maintainability. <http://estep.esciencecenter.nl/> .

- ✓ Problem description:
The Netherlands eScience Center operates as a bridge between science and IT and e-infrastructures in order to "accelerate discovery". The software that is developed because it is not readily available elsewhere, is required to be fit for purpose, re-usable, excellently written, practical and maintainable. To this end, rules are being designed and developed, also taking into account that NLeSC is not necessarily the original designer of the codes. These types of rules of conduct are rarely adoptable from elsewhere, whether in industry or science, so NLeSC is learning on the job.
Managing a programme such as eSTeP is complicated in that it is concerned with a huge variety of codes in many languages for many different purposes. Within the projects that NLeSC is involved in, most of which are (co)funded by NLeSC, software development and adaptation to new use cases are the main contribution from NLeSC to the project.
- ✓ Software engineering:
Software in the domain of interfaces (in the most general sense of the word) is mostly developed in HTML or Java Script. Software for data analysis is mostly written in Python and scientific codes are dominantly written in C, C++, Fortran or sometimes Cuda.
Doing so, generally accepted guidelines for framework use and engineering approaches are implemented and used wherever possible. For typical Fortran or Cuda environments, such frameworks do not exist yet. There, best practices in programming and development are followed. NLeSC hardly develops for Windows or Mac OS-X systems. Linux and Unix are the standard operating systems under which the codes operate. Where applicable, Web interfaces allow usage from Windows or OS-X.
The development environment is typically GIT-based. It is used for version control (unless an older version control system was in use already), open git repositories, github for licensing of the (generally) open source code.
Whenever possible, NLeSC tries to use or re-use existing software and tools and tailor it to the application required for the research project at hand. But very often, new software needs to be created. Then there is always the question of balance: how to meet the requirements for both speed and efficiency and modular design with a view to re-usability and maintenance.
- ✓ Software distribution:
Distribution is done via the Web: each code has its Web domain (mentioned on the Git server), available through cloud services, from where the packages can be downloaded in a format suitable for the particular language used. Binary packages for Fortran, C, C++ codes, if required packaged into containers, and other standard packaging tools for Java and Python.
- ✓ Organisation:
All software developed or adapted at NLeSC plays a role in cooperative projects with "external" parties, except for a small fraction of the eSTeP codes which have a general purpose character (for example methods to link different resources over the Internet in a simple manner). As NLeSC works mostly on the basis of granted projects, the organisation is able to define principal conditions for software development in the granting conditions. For the rest, this is a matter of negotiations with the applicant.
Internally, NLeSC has defined strong regulations regarding the way software is written, versioned, documented, and conceptualised in terms of modularity and performance. Although this is not a formal internal directive, NLeSC seeks to maintain industry standards to the industrial strength software it creates.
- ✓ Referencing:
All software plays a major role in (scientific) publications, because that is the very reason the software is being created in the first place. Use of eSTeP software and tools must be cited in all publications. This includes the name of the code, its version (release) and DOI (Digital Object

Identifier), with its persistent link. The documentation with the software describes the way the software needs to be referenced (can be found through GIT or Zenodo). The various software releases may or may not be backwards compatible. However, through the Git system all previous versions remain available. If so required, they are put into a container that enables using the code in a virtualised older system environment.

✓ Users:

Most (co)applicants of projects are at the same time the major software users. As a rule, being co-developers, they set the minimum requirements for the scientific goals, the user interface and the user requirements and are the principal sparring partners for quality, effectiveness and user friendliness of the software and tools. No special testing regime applies to make the software fool proof, as usually this is not an *a priori* requirement. Code cracking sessions are under consideration at NLeSC to test the fool-proof score of the code.

The user groups are generally not differentiated into beginners/advanced/expert levels. However, after the first operational tests involving the principal investigators, more levels of users come on board and adaptations can still be made to the code during the project phase. The rest will have to learn through the extensive documentation facilities.

✓ Documentation:

The documentation is considered an essential part of the software. The git environment, the website with all its demos, documentation and specifications serves as extended documentation. Furthermore, screenshot-supported YouTube/Vimeo movies are available to demonstrate software installation and use.

✓ Sustainability:

The principal investigator (the applicant of the project) is considered the software owner for all practical purposes (even though it is mostly open software). He generally is a member of a larger community which is also served by this software. NLeSC feels responsible for delivering optimum ease of maintenance and future sustainability, but cannot formally be held responsible for any future maintenance. Bug fixes and similar issues are a matter of good will. Requiring future maintenance of the software to some degree to be agreed before project start is under consideration as a granting precondition, to be signed by the applicant.

6.8 Use case: Tundra, treebank repository and search tool

Author: Dirk Roorda

Place, Date of interview: Tübingen, 23-2-2016

Discussed with: Alexandr Chernov (University of Tübingen), developer of Tundra

Marie Hinrichs (University of Tübingen), researcher

Abstract

A treebank is part of a standard linguistic tool chain from text processing through parsing to semantic analysis. Treebanks represent the results of syntax parsing. They are being used to visualise sentence structure and to facilitate queries on particular sentence structures. See <https://weblicht.sfs.uni-tuebingen.de/Tundra/>. To enter Tundra you have to go through a process of authentication via your home institution or CLARIN, which is identical to accessing WebLicht (see: https://weblicht.sfs.uni-tuebingen.de/weblichtwiki/index.php/FAQ#How_do_I_log_in_to_WebLicht.3F).

✓ **Problem description**

The immediate reason to undertake Tundra was that an older treebank tool, Tiger Search (<http://www.ims.uni-stuttgart.de/forschung/ressourcen/werkzeuge/tigersearch.html>), was becoming obsolete. The construction of Tundra can be seen as an act of software sustainability: it replaces obsolete software with new software, though not in a backwards-compatible way. [The query language is preserved (with minor changes) so that Tiger Search users can switch to Tundra without learning yet another query language.]

Syntax parsing of natural language is mostly an automated process, but human decisions are frequently needed, partly depending on the treebank – some are annotated automatically, some fully manually, and some use a combination. Extracting sentences with particular structures is a typical information retrieving task.

People can use the tool to search treebanks for particular structures and/or gather statistics on those structures. Tundra is a treebank system, developed at the University of Tübingen within the wider context of CLARIN-D, which is the German department of the CLARIN ERIC, the legal

organisation that represents the results of the FP7 project CLARIN (Common Language Resources Infrastructure and technology). In terms of funding, the project presently represents a dedicated researcher for three years.

✓ Software engineering:

The software is written in Java, Javascript and XML. The software system is cross-platform, but developed in Linux. The development environment was a local git (GitBlit), not github. Github is scheduled for when the software is more mature. Currently there is only one version of the software, with a new version under development which focuses on improvements in modularity (separating frontend and backend), performance, and presentation (better look and feel through use of more modern Javascript libraries). For the Web application itself, backwards compatibility is not an issue since the query language and execution remain unchanged.

The Tundra treebank format is highly flexible to accommodate all of the annotations that various treebanks contain. This does not affect backwards compatibility, however, since a new version of a treebank does not replace an older version, rather the new version is added as a separate treebank. There are currently no plans to modify the treebank deposit format.

The trees in the treebanks can be linked individually, but at the moment there are no particular measures in place to keep those links stable. As long as the treebanks themselves are available, it will be possible to make queries and retrieve trees.

A detailed dependency scheme is available:

- ZK 6.5.2 CE: Open source Ajax framework for Java-based Web applications.
- BaseX 7.7.3: XML Database engine and XPath/XQuery processor.
- Jetty 9: Open source embedded Web server and servlet engine.
- JavaCC 5.0: Java parser and compiler generator.
- Bootstrap 2.3.2: Frontend for Web development.
- jQuery 1.9.1: Cross-platform Javascript library.
- canvg: HTML5/Javascript canvas tools for SVG graphics.
- Mouse Trap: A library for handling keyboard shortcuts in Javascript.
- Roboto: Google Web font licensed under Apache 2.0.
- Gentium: Summer Institute of Linguistics Web font licensed under SIL Open Font License.

The current production version is rather monolithic. The code is being refactored into modular pieces which can also be used by other projects (e.g. the frontend will be used for visualisation in WebLicht in the future). It has a backend with a REST-API for retrieving treebanks, and a Web frontend. This is in contrast to the tool it is built to replace, Tiger Search, which is a standalone tool with a graphical interface that must be installed on the user's computer. The division into frontend and backend will make future changes easier.

Everything is built from scratch, but Tundra is used by Weblicht (<http://weblicht.sfs.uni-tuebingen.de/weblichtwiki/>) for tree visualisation and searching. CLARIN functionalities used are:

- CLARIN authentication (<http://clarin.eu/content/federated-identity>)
- Helpdesk button (<http://clarin-d.de/de/hilfe>)

See also the open source libraries mentioned above.

✓ Software distribution:

It is a Web application which is not being distributed for two reasons. It is being heavily developed right now. And it is meant for an institution that serves a community, not for individual end users.

✓ Organisation:

The project was mainly an in-house development, close to the people who also determine the requirements. Requirements collection seems to be very much an ongoing process. For the rest it was basically an incremental approach.

Legal issues were to be addressed for each treebank.

✓ Referencing:

A journal article by Prof. Erhard Hinrichs, soon to be published, uses Tundra extensively.

The software is citable by proxy of a journal article: Scott Martens (2013). TUNDRA: A Web Application for Treebank Search and Visualisation. In: Proceedings of The Twelfth Workshop on Treebanks and Linguistic Theories (TLT12), Sofia, pp. 133–144. URL:

<http://bultreebank.org/TLT12/TLT12Proceedings.pdf>.

Currently, results can be reproduced by simply re-running the query. Use of PIDs for query results is under consideration.

✓ Users:

Ideas for enhancements have come from the Help Desk ticketing system as well as from researchers using the software. Easy access with single sign-on for academic users, performance and a flexible format for treebank import were the most prominent wishes.

The target audience is made up of language researchers and philologists, and other digital humanists. The treebanks in the repository give an indication of the disciplines involved:

- Kafka: Der Prozess
- German newspapers (85,000 texts)
- Tübingen Treebank of spoken German (38,000 sentences)
- Index Thomisticus (15,000 sentences from three works of Thomas Aquinas)
- Corpora in English, Bulgarian, Japanese
- German Wikipedia.

The software is also aimed at institutional users, administrators, infrastructure providers.

Treebanks date back as early as the 1950s, so they are well-known among linguistic researchers.

There is active usage. The main point of contact is the help-desk.

✓ Documentation:

There is an in-page tutorial. When the software is fully in production, courses for end-users will be planned. Extensive documentation and tutorials are available in the application itself

(<https://weblicht.sfs.uni-tuebingen.de/Tundra/>). End user help (user guides and tutorials) is available. Treebanks can be deposited into the CLARIN-D repository in Tübingen

(<http://www.sfs.uni-tuebingen.de/ascl/clarin-center/repository.html>).

✓ Sustainability

General Computational Linguistics Staff at the University of Tübingen is presently maintaining the software, which is being further developed right now, the top priority being to realise all of the desired functionality. Currently, development and maintenance are tied with CLARIN-D. That will end soon, but is expected to be extended. If nothing comes of that, the current staff feels it is their responsibility to keep the system alive. They experience a keen interest from their user community. The software is formally owned by the General Computational Linguistics department. Future sustainability is not a priority in the current stage of development. The software is being mirrored at the Garching computing centre, and another host is being planned. But these concerns have to do with day-to-day reliability rather than long-term usability. How the results obtained by means of the system can be made available for the long term is being considered. been implemented.

6.9 Use case: Gaming software

Author: Patrick Aerts

Place, Date of interview: Virtual interview Hilversum, Beeld&Geluid, 16-2-2016

Virtual interview with: Eric Kaltman, previously computer game architect at Stanford University and researcher at the University of California, Santa Cruz.

The interview report is based on a presentation at Beeld&Geluid given by Eric Kaltman and written material he subsequently provided to the author for the purpose of this use case.

Abstract

Preserving games is a complex business that goes well beyond keeping the software executable. In the US a few institutions keep large collections of games, some dating back to the early days of computing. The problems and issues related to keeping the software running under modern conditions are presented together with some guidelines.

✓ Problem description

Computer-based games have been around since the early sixties. While originally such games were implemented on the very type of computer systems in use for other, general, purposes, in due course an industry arose around computer games introducing special devices and equipment, while other games remained executable on laptops and more modern handheld devices, including mobile phones. Keeping all these different types of games alive in a way that enables people to at least understand what the fun was all about is a very complex matter, if only because the industry, in order to protect itself and its investments, hardly shows any interest in games preservation. Indeed, the industry tries to speed up obsolescence of software and devices as to be able to sell newer versions to keep the money flowing. Yet the development of games provides an enriching view of cultural history as they represent an image of the time in which they were developed and deployed.

✓ Software engineering:

Games software has been written in all sorts of languages, from C to Javascript, and possibly even Fortran and assembler. Modern computer games are written modularly, just as in most modern computer codes. Versioning of games software is often pretty obscure, developing with the device types brought onto the market and with operating system updates and upgrades as customary on mobile phones and other handheld devices.

✓ Software distribution:

Depending on the origin of the software, various distribution schemes exist. The codes may be open software, mostly developed at universities and research institutes, for example for educational or other serious purposes (serious gaming), but games are also distributed through physical stores or by downloads from the Internet or App stores.

✓ Hardware platforms:

A typical aspect of games compared with other forms of software is that the device on which the software runs or for which it was specifically designed plays a major role in many games. Nintendo, Xbox, etc. typically are based on first selling the device and then selling different games that can only be played on that specific device and perhaps on its next hardware release, but usually not beyond that, in order to promote the sales of new devices and software. This is a typical additional aspect of games conservation beyond the software sustainability aspects *per se*.

✓ Organisation:

With some exceptions, games are designed in teams, although there have individual designers too, both in the very beginning and in present times (Flappy Bird). Companies may grow extremely fast due to a successful introduction of a game and may disappear just as quickly if no new concepts come to mind to sustain success.

✓ Referencing:

In the US, metadata formats and collection coding systems have been and are being conceived, but no international standards presently exist.

✓ Users:

All games have (or had) users. Some games even have large and strong communities. These communicate beyond the domain of the game on all issues related to gaming. There are even professional gamers, particularly where money is at stake. Modern games are tested first in small test-user groups. A modern complication for gaming software sustainability is the fact that games may be played over the Internet, involving one, a few or many users at the same time, which act as opponents or co-players. Even children can play adventure or creative games together, each using their own device (usually handheld).

✓ Documentation:

As far as we know there is hardly any publicly available documentation of games software. By contrast, there is documentation about the way the game should be played, but increasingly so as part of the game or the gaming process itself (for example in adventure games). A Help button is the most obvious, today even “clumsy” way of providing background or directives. Sometimes indications for next steps in the game can be earned throughout the game. Some games are by nature adventurous and no documentation exists. That this can work nonetheless can be derived from the many very young children that seem to be able to play a game that does not even communicate in their own language or in a language they cannot (yet) read. It is not known if games software producing companies keep proper documentation at all.

✓ Referencing:

Gaming software citation is complex and no standards have been adopted yet. One can distinguish between citing (and linking to) the software proper and citing the emulator environment in which the software might run. The complexity arises from facts such as that the software resided on a cartridge of a certain (physical) format, with its own version and specific characteristics (some cartridges contained extra chips for additional processing capabilities) in addition to the software versioning and the versions of the main device (think Nintendo). One serious document on this topic can be found at <https://gamecip.soe.ucsc.edu/about>.

✓ Sustainability:

The nature of software is such that with some effort it can usually be made to run on any computer that has the minimum specs (memory, processor speed, etc.). This is probably true for much software, but here it extends to at least gaming software. One can now define a virtual machine in a cloud environment and/or access an emulator, which has also become available more often. Emulation involves translation of instructions from one platform to another. This requires computational performance beyond the original device performance, all the more so if tricks have been implemented in the game software, such as playing around with the difference in speed of a processor and a co-processor in the same device. This way a MHz-scale device may require a multi GHz processor to do the emulation right, i.e. to make the game playable. Another type of problem arises when a game involves getting access to some proprietary software (owned by the game copyright holder/creator), either as part of the game process, or to verify the legitimacy of the player’s ownership/rights (authentication). And then there are games which are community based. Even if one is able to emulate the game, one does not get the experience back. Games that are downloadable, in particular on a mobile device (mobile phone) are disastrous as far as sustainability is concerned. Versions are strictly linked to the mobile OS versions. You cannot save the game (or get rid of it, once installed).

Extracting software from hardware media: an example of an extensive effort is that of a collaborative project by Stanford University and NISC (the National Institute of Standards) under the name of Cabrinety²⁷, which creates forensic disc images and high-resolution photographic scans of materials in the software series of the Stephen M. Cabrinety Collection in the History of Microcomputing, ca. 1975-1995. It involves software from dozens of media formats (such as floppy disks, computer cassettes, and game cartridges). For this purpose, custom made reading devices have been made.

An authoritative publication on games preservation is “Preserving virtual worlds”²⁸ by Jerome MacDonough et al. dating back to August 2010. It is still one of the best contributions to the discussion in this domain.

²⁷ See <https://library.stanford.edu/projects/cabrinety-nist-project>, <https://library.stanford.edu/blogs/stanford-libraries-blog/2016/04/preservation-week-5-questions-charlotte-thai>

²⁸ http://mith.umd.edu/wp-content/uploads/2012/03/whitepaper_PVW.FinalReport.pdf

6.10 Use case: HSN (Historical Sample of the Netherlands)

Author: Peter Doorn (DANS)

Based on information from and discussed with: Kees Mandemakers, IISH/EUR
(<https://socialhistory.org/en/hsn/about-hsn>)

Time frame: 1987-2016

✓ Abstract:

The Historical Sample of the Netherlands (HSN) is spanning 30 years of construction, development and maintenance of databases and related software. The HSN tools and data serve to reconstruct life histories as completely as possible for a representative sample of the nineteenth- and twentieth-century population of the Netherlands, drawn from all persons born between 1812 and 1922. The HSN database will include individual data for 77,000 persons, including items like age at marriage, religious affiliation, number of children born, occupation, place of birth, literacy, social network and migration history.

✓ Software and development environment:

The software has been developed, maintained and adapted since 1987. In the late 1980s and early 1990s data entry programmes were developed in dBASE (a popular dbms package for MS-DOS) and Clipper (an xBase computer programming language and compiler) to make historical population records machine-readable. Output routines and reports were developed to link and present the information in the various database tables. When Windows overtook DOS, these tools were replaced by MS Access and SQL queries. Later still, the HSN tools were migrated to Delphi, a Rapid Application Development (RAD) tool which evolved from Borland's Turbo Pascal for Windows. This change was not long-lived, and the data entry software then became Web-based, with separate applications/web forms for each historical source.

Routines have been developed to check the data for completeness and consistency, to link data and to identify individuals. Those routines have been written in Java, a choice based on the software policy of the Institute made six or seven years ago. However, the institutional software policy tends to change over time, leaving the researchers with legacy software that after some time is no longer supported by IT staff.

✓ Domain software specifics:

Originally the database design and tools to operate and manage the data were developed by the HSN Principal Investigator (PI) Kees Mandemakers himself, but in the course of time and due to the changes in technology this became gradually unfeasible, and the software development/maintenance task was taken over by the department of digital infrastructure of the International Institute of Social History (IISH, in Dutch IISG). Currently the software is again being renovated. The increased distance between researcher and tools is experienced by the PI as having advantages as well as disadvantages.

One of the important changes in the course of time was the implementation of the concept of what is called the Intermediate Data Structure (IDS), a format for exchange and integration of historical life-course data adopted by the main international longitudinal population databases in Europe and the US.

Data integration is one of the more complicated tasks of the software. From the IDS a general data release and customised database tables are made, for example in STATA (Data Analysis and Statistical Software) and R (another software environment for statistical computing and graphics).

✓ Software dependencies:

The researchers feel that they are dependent on the choices made by the IT department of the institute, and sometimes they question whether the choices made are helpful for keeping their software tools alive. Decisions to support some database systems or programming languages rather than others may implicate that the expertise/support needed to maintain older software can suddenly evaporate. Mandemakers has experienced the lock-in caused by choices made in the past the hard way. He feels this lock-in is especially problematic when depending on others. Sometimes the updates are technology-pushed rather than required by the user/researcher. The latest trend is to make more use of Python, although Mandemakers would prefer to change as little as possible.

Data security is also an example of a need that is created by certain institutional decisions. Now that data-entry is taking place via the Internet, the data protection requirements have increased, which is costly for the project.

Sometimes old machines and operating systems are used, particularly PCs running DOS, because a software routine has not been upgraded to new machines and OSs.

The costs of software development and maintenance have never been separately calculated or administered. Currently 0.5 FTE is working for 1-1.5 years to convert the data-entry software; additionally, a developer has been working for 3 years on the output software (including checking the consistency/validity and conversions); Mandemakers estimates the required maintenance work at 10 hours per week.

✓ Software re-use, distribution and referencing:

The software has always served internal use only (data entry, cleaning, consistency checking, linking, management, presentation). There is no distribution of the software itself. End users import the HSN data into their own tools, which they use for further analysis. The software is therefore generally not open source. The IISH has however recently implemented a new policy of storing the software developed at the institute in Github, see: <https://github.com/IISH>. The homegrown HSN software is stored on servers of the Institute itself.

The HSN data are distributed in a series of releases (in DBF or CSV format). Sometimes data inconsistencies are found that have not been detected earlier; correcting them complicates the replication of results.

✓ Organisation:

HSN is a foundation hosted by the ISSH, which has been funded by a range of successive projects, with minimal sustainability guarantees by the institute. The HSN software policy is necessarily *ad hoc* because of the scarcity of financial means.

✓ Users:

The user community of the HSN consists of social historians, historical demographers, and other researchers interested in historical population data, including health care and other life sciences.

✓ Sustainability aspects:

The driving force behind HSN is Kees Mandemakers; his biggest worry is whether a successor can be found to provide continuation. It is doubtful whether the Institute can take over the responsibility without the drive of an involved researcher.

Without the software, the integrated HSN data will be impossible or very hard to use. New software would have to be written to make the data usable again. Although extensive and detailed documentation is available, describing all data processing at the various levels, the development of such software would require a considerable effort.

The individual source files are accessible in raw (checked and corrected) form, but links between persons in different sources and their mutual relationships will be lost without the software.

6.11 Use case: Preserving sound and vision

Author: Patrick Aerts

Place, Date of interview: Amsterdam, Netherlands eScience Center 21-4-2016

Interview with: Jesse de Vos, researcher at Beeld en Geluid, Hilversum, the Netherlands

Abstract

Beeld en Geluid (B&G) holds all broadcasting information, digital and analogue, of the Netherlands broadcasting organisations. It has also acquired a large collection of games of Dutch origin. B&G offers services to the public broadcasting organisations and through them also to the general public. B&G services involve a variety of software products, of which the sustainability is not always a big issue. Issues on the horizon mostly involve the future of Web services.

✓ **Problem description**

The collection of B&G includes physical (music and speech) records of different making, sound collections on records or tapes of many formats, photographs, film, video in various formats, archives, websites and some Web archives. The additional concern for games preservation is due to B&G's general knowledge in the domain of graphics and sound, which is more extensive than the National Library's. The B&G software collection is the sum of customised off-the-shelf packages, homegrown and open source software. Software is involved in the copy/reformatting workflows to make useable digital versions of all physical digital and analogue sources, to make those new sources accessible and findable and to deliver products on demand to broadcasting organisations. B&G experiences no further specific problems, except for bit rot, as they still have most of the original physical equipment on site to feed the workflows from the origin and most modern software tools can be applied to make the best copies in today's formats. The collection is extensive: about 25 Petabytes (including copies).

✓ **Software engineering:**

There are no strict software engineering rules in place, and writing homegrown software from scratch is not part of the core business, but is done when nothing is readily available from elsewhere. Where applicable, emulation as a service is used to run older pieces of software, or software is made to run again by invoking Javascript and use the browser. B&G does experiment with Web archiving, which is considered the next big issue.

✓ **Users:**

There are three types of users: internal users (employees), the broadcasting organisations and the general public. None are actually involved in any testing of homegrown software products.

✓ **Sustainability:**

Software sustainability is not seen as a primary issue for B&G. There are concerns about the preservation of the underlying, mostly analogue materials, which suffer from deterioration. So a best effort is made to make digital copies of all the analogue material in formats that will be accessible for future software programs. In addition, the original materials will be kept under the best possible conditions.

An exception to this general practice is the games collection, which is discussed as a separate use case.

6.12 Use case: TSTC, software to manage Dutch language data

Author: Peter Doorn (DANS)

Based on information from: Remco van Veenendaal, Griet Depoorter, Martin Everaert (<http://tst-centrale.org/>)

Time frame: 2004-2016

Abstract

The *Taal- en Spraak Technologie Centrale* (TSTC, Language and Speech Technology Centre) maintains software to manage a collection of Dutch language data resources of national importance. TSTC is the knowledge and distribution centre for Dutch text collections, (scientific) dictionaries, speech corpora and language and speech technology software. Those materials include results from grant programs such as STEVIN, outcomes of PhD research and items developed by organisations such as the Institute for Dutch Lexicology (INL). TSTC collects and

manages these products and makes them available for reuse in education, research and development. TSTC was founded on the initiative of the Dutch Language Union (*Nederlandse Taalunie* - NTU) in 2004.

TSTC was hosted by INL from 2004 to 2012, and by NTU from 2013 to March 2016. Since April 2016, INL has again been the host, with the intention of integrating the facility into the core services of the institute in a new form, as *Instituut voor Nederlandse Taal* (INT).

✓ The software:

The product overview of TSTC in March 2016 mentioned 114 products, 12 of which were categorised as "tool". However, several of the other products (corpora, databases, dictionaries) require specific Web applications or other software to access them. An example of the latter category is the Dictionary of the Dutch Language (*Woordenboek der Nederlandsche Taal*). In April 2016, the TST catalogue (again under the responsibility of INL) listed 107 materials, 7 of them tools. The characteristics of the tools are reflected in a table, based on the documentation available from the TSTC catalogue.

✓ Domain software specifics:

Every tool in the TSTC software catalogue²⁹ has its own specifics, depending on the original group of developers. The same is true for the availability of the software and its documentation. See the "Software" row in the table²⁹.

✓ Software dependencies:

Software dependencies vary from tool to tool. Some tools require a Windows PC and specific Windows versions; other tools are available for Linux or DOS, or Debian or Ubuntu. See the "Operating system" row.

✓ Software structure:

This is different for every tool, further details can be found in the documentation, a link to which is provided for most tools in the "Documentation" row. One tool uses Github for documentation.

✓ Development environment:

See the "Software" row in the table²⁹.

✓ Software distribution:

The software tools are aggregated and distributed via the ordering system of TSTC. There are references to the original project websites. In March 2016, some of the links no longer worked (link rot). Those tools have now disappeared from the product list. This makes the problem of sustainability painfully clear.

✓ Re-use and development:

The projects were finished between 2007 and 2014 (see "Year" row). The TSTC website does not provide information on reuse statistics of the tools. All tools except one are freely available. One has a non-commercial license of 500.

✓ Organisation:

Before 2013 a team of about four people was responsible for the maintenance of the products and services of TSTC. In March 2016 just one member of staff was responsible for the upkeep, with the additional support of an external company hosting the servers and storage. Since April 2016, the development team of INL has been responsible.

✓ Users:

There are three target groups who most frequently avail themselves of the TSTC services: researchers, companies and the general public.

- The products distributed by TSTC are mainly used as research materials by scholars in the field of Dutch language, linguistics, and language and speech technology. Students make use of the materials, usually through their faculty or research institute.
- Technology companies use the products to deal with commercial applications. Examples include spelling checkers and voice telephony.
- Others, i.e. anyone working with source materials regarding the Dutch language, professionally or out of interest. Most popular in this category are the online dictionaries such as the Dictionary of the Dutch language (WNT).

✓ Sustainability aspects:

As a result of the project agreements, the ownership of project results was usually transferred to the Dutch Language Union (NTU) at the end of the project. When ownership is transferred to NTU, TSTC becomes responsible for the resource's lifecycle.

Although TSTC is not a software company and has a finite budget and capacity, several software packages are taken care of and are being maintained. Within NTU this was based on ITIL best

²⁹ See table in appendix 9

practices. As it is not possible to maintain all software resources all the time, choices have yet to be made. One possibility is to distinguish and choose between minor and major maintenance. Another option is to back up (open source) communities via some form of "escrow". These possibilities at least provide the assurance that someone will (be prepared to) look after the sustainability and lifecycle of certain software packages in the future.

The ITIL-based lifecycle model for software sustainability includes the following:

- Management (archiving) keeps software available as is it.
- Maintenance keeps software usable or improves it.
- Making software available gives users access to software.
- Supporting software shortens learning curves.

TSTC distinguishes between minor and major maintenance, which is one solution for the problem of having a finite budget and capacity – choices are made based on use, user feedback, severity of bugs, complexity of the software and availability of time, skills and money.

- **Minor software maintenance** - The aim of minor maintenance is to keep resources usable, which means fixing critical bugs, updating manuals and documentation and upgrading the software to work on newer versions of operating systems or with newer versions of programming languages. Minor maintenance is done by TSTC or their maintenance partners. Periodically, TSTC checks if resources require minor maintenance and starts the work after having consulted the owner – often NTU. Feedback from users is included in these maintenance checks. The result of minor maintenance is usually a patch or update of a resource. News on any updated versions is published, so that users can request an update.
- **Major software maintenance** consists of significantly improving or expanding a resource. It usually requires additional funding and collaboration with developers and external experts. Information and advice on which resources should be improved or expanded is gathered from the various advisory committees that assist NTU and TSTC and from user feedback collected by the service desk. From this information it is possible to gather how urgent and/or important the major maintenance is. Major maintenance usually results in a new version of a resource rather than a patch or update. News on any new versions is published and users must accept a (new) license.
- **"Escrow"** - In some cases, software is not handed over to the community, but still to some extent looked after by TSTC. Agreements with the communities that developed the software packages ensure that if the community stops supporting the software, NTU will take over the responsibility for the software's future lifecycle. This agreement is not an actual source code escrow, but has some aspects of it and at least ensures future management and maintenance of the software.

✓ Referencing:

The website of TSTC is <http://tst-centrale.org/nl/>.

In the table²⁹ there are references to the individual software tools

6.13 Use case: Digital archaeology – *De Digitale Stad*

Author: Patrick Aerts

Place, Date of interview: Amsterdam, 14-4-2016

Discussed with: Tjarda de Haan, e-archaeologist at Amsterdam Museum

Abstract

De Digitale Stad ("The Digital City") was an early nineties project (1994-2001), set up by an Amsterdam collective to create a virtual city that mimics real city life, or in other words, using the city as a metaphor to structure and shape the then rather unknown cyberspace, using the most modern techniques for messaging, communication, information sharing, focused on the city of Amsterdam. Initially, it was financially supported by the city of Amsterdam. However, at some point in time in 2001 the project was discontinued. Because the project reflects many aspects of digital life at the time there is strong renewed interest in all aspects of the former project and people have started diving into what is left of it. The tapes, servers, codes written, etc. have to be found back, retrieved from all dispersed sources and made to work again.

✓ **Problem description and background**

The Digitale Stad project was conceived by many fathers and mothers, but the best description of their involvement would be to call them a "collective". At the heart of the collective were the cultural centre *De Balie* and a periodical entitled *Hack-Tic* (now known as the Internet provider XS4ALL). Among the driving forces was Marleen Stikker. The group succeeded in getting funds from the city of Amsterdam, twice €50,000, and some additional funding from other sources. Much of the work, however, was done by volunteers. It was the first-ever free for all Internet experience in Europe. The problem today is that since 2001, when the endeavour ended, all of the materials that made the Digitale Stad operate as it did got dispersed, devices have become obsolete, software got lost, tapes drift around at various places, data are unknown, link rot and bit rot, etc.

The Amsterdam Museum started a pilot project in 2010 to safeguard the Amsterdam digital heritage because this unique heritage, and especially the digital memory of the early Web, is at risk of being lost. Or worse: already gone. The aim is to answer the questions, involving many partners: how to excavate, reconstruct, present, preserve and sustainably store born-digital heritage and make it accessible to future generations? The project is loosely called digital archaeology, because getting the old stuff together requires a lot of tracing, physical, social, technical and digital as well. One tape drive was in use as a foot step under a desk, tapes are being found in cellars and attics, snapshots of websites are being found in Web archives, etc. The ultimate goal is to get it all working again and display it interactively as a piece of digital history of Facebook-type dimensions.

There is a follow-on activity hosted at dds.nl: *De echte Digitale Stad* ("the true digital city").

Further details about Digitale Stad can be found at https://nl.wikipedia.org/wiki/De_Digitale_Stad (in Dutch).

✓ Software engineering:

De Digitale Stad is a complex information system with various applications linked together. It is interactive, internally networked and process and context aware. It used a telnet interface, static HTML (2) and interactive Web pages. The operating system was Unix. All tools and designs were highly sophisticated at the time, but hardly any records were kept, no documentation was made, there should have been "freezes" but there are not.

✓ Software distribution:

The software as such was not distributed. Everything ran on central servers, with tape drives and communication servers directly attached. Users had to log in, using telephone lines and modems. The rest could be done through the Web interfaces.

✓ Organisation:

The creators worked together with the local "anarchistic" hackers' collective, which may give an idea of the level of organisation at the time. Probably the description "just enough organisation" would do right to the effort, because after all things worked properly, certainly taking into account the level of funding. Of course, it makes the archaeological efforts required to get things working again all the more challenging. On the other hand, a foundation, "Stichting De Digitale Stad", was set up in 1995 to provide the funding parties and other governmental organisations with a formal party to do business with.

✓ Users:

Basically, everyone was the audience. The heart of the activities was in Amsterdam, but anyone

with an account could participate. Users were called "inhabitants" who could meet up in cafés, squares, streets, groups. The general public was the audience, which was ground breaking at the time: Facebook *avant la lettre*.

Users were not as such involved in testing but the response would guide the designers towards improvements. The number of users grew relatively fast, given the fact that telephone bills were still enormous, compared to today: 1994: 12,000; 1997: 60,000; 1998: 80,000; 2000: 140,000.

✓ Documentation:

Communication and information were considered important for the quick proliferation of the concept and its use. The national Ministry of Economic Affairs even ordered and financed a book entitled *Handboek Digitale Steden*³⁰ ("Manual for the development of Digital Cities"), which was actually used by other cities than Amsterdam to set up their own cities' digital versions.

✓ Sustainability:

Obviously, in this case the lack of sustainability made it necessary to set up an archaeological quest to find back all the necessary materials, the programs and HTMLs, the equipment on which the software and peripheral devices worked. As sometimes happens, people took other directions in life, were interested more in creating than in maintaining, while at the same time access to the Internet as such became much more affordable for the general public. Anyway, De Digitale Stad ended pretty suddenly.

Follow the project:

- <http://hart.amsterdammuseum.nl/re-dds>
- <http://hart.amsterdammuseum.nl/23dingenwebarcheologie>.

"The reconstruction of The Digital City, a case study of Web archaeology", by Tjarda de Haan and Paul Vogel: <http://www.bitsandbytesunited.com/?portfolio=publication-the-reconstruction-of-the-digital-city-a-case-study-of-web-archaeology>.

³⁰ Kees Schalken, Joost Flint, *Handboek Digitale Steden*, January 1995.

6.14 Use case: Born digital art – SHFT-34

Author: Gaby Wijers
Concerns time frame: 1982-2007
Discussed with: Peter Struycken

Abstract

This case concerns the process that the SHFT-34 program went through from its origin to its present form. It is an historic form of digital art, formerly owned by the *Groninger Museum*, but the original code got lost. The case describes the preservation route and the issues faced on this journey, which may be a prototype of other similar challenges in digital art preservation.

✓ Problem description and background

Peter Struycken ([Den Haag, 5 January 1939](#)) is a Dutch figurative artist and colour expert. He started producing computer art as early as 1969.

SHFT-34 is a version of the SHFT program. Different sets of parameters determine its behaviour. Peter Struycken has produced other works that include the name SHFT followed by a number. Examples are "SHFT-30" and "SHFT-31" (1982). SHFT works can also be photographs, paintings etc. on which the computer program was used to extract stills.

SHFT-34 is a computer program written between 1981-1982 in the Pascal programming language. The work was created for permanent display in a coffee corner of the Groninger Museum. Due to a reorganisation of the museum, the permanent positioning was postponed. After the reorganisation it turned out that a vital part of the machine had gone missing; it was never found again. Only ten years later, SHFT-34 was found on the computer of a retired technical staff member of the museum. Thanks to this discovery the work could be exhibited again, but with the standalone machine missing, not in its original setup.

Several attempts were made to make the program run again.

In 2006 an attempt was made to run SHFT-34 on a modern computer. Technology had advanced considerably since the conception of SHFT-34 and it did not work on the prevalent systems. In 2007 Floris van Manen reprogrammed SHFT-34 to Java. Whereas the hardware in the 1980s held a lot of restrictions in terms of speed and overall possibilities of the program's functioning, contemporary technology in 2007 allowed for enhancement. Van Manen and Struycken did make some changes that according to them improved the work while staying true to the authentic version. The changes consisted of two different view modes and a click-and-drag option which allows the viewer to "move" through the colour space by hovering the mouse. The interactive options in the reprogrammed version help the viewer to better understand the colour space as a whole.

✓ Software (re-)engineering:

There were no versions of the original code in existence, so the SHFT-34 software was rewritten in Java, although Java is rarely used by artists. Java runs on most operating systems among which the most popular ones: Windows, MacOSX and Linux. So SHFT-34 with Java Runtime Environment is OS-independent. Today SHFT-34 also runs on Java 8 update 91 (2016).

Because even in Java there are no guarantees that the software will keep running on ever newer technologies, sustainability methods were explored to freeze the present running environment. An open source operating system for SHFT-34 that also runs as a virtual machine is Debian 7. For this open source all versions remain available.

Furthermore, the virtual machines VMware and Virtual Box were tested under Windows, MacOSX, and Linux (Debian en Ubuntu). On all these versions SHFT34 kept on working.

As a result of this research, LIMA was able to design a workflow for intake and handling of digital/software based art. These and related findings will be published in the summer of 2016. At the time of development, no development environments other than text editors were in existence.

✓ Distribution:

The art work has a limited scale distribution on CD-ROM. On the CD-ROM the SHFT-34 program is distributed together with Java JRE for Windows, MacOSX and Linux. A double click is sufficient to start it up, but in the OS the system can also be made to auto run.

✓ Organisation:

At the start of the reprogramming project, there was close cooperation between Peter Struycken and Floris van Manen. For the virtualisation Paul Jansen Klomp came to assist them.

The broader context was the *Transformatie Digitale Kunst* project with the *Behoud Moderne Kunst* foundation.

- ✓ Documentation:
The use of SHFT-34 is described in a pdf on the above-mentioned CD-ROM. The documentation is rather extensive and an interview with the artist is available at LIMA.
SHFT is fairly "fool proof" because as an art object it was designed to be open to the general public. Basically, it is available for anyone who cares to buy it. In practice this means museums and art collectors.
- ✓ Users:
The software is well appreciated by the users (basically the general public).
- ✓ Ownership:
The artist, the programmer, the conservation manager and all who bought the object consider themselves owners of the artefact.
- ✓ Sustainability:
The art work in its present form (in Java) is made for sustainability. The formal owners (*Rabo Kunstcollectie* and *Kröller-Müller Museum*) have handed the conservation task to LIMA.
The workflow resulting from the project can and will be re-used.
The cost of the conservation project will be some €10,000 excluding hardware, software and other equipment.

6.15 Observations from the use cases

The use cases presented are of course just snapshots of individual events, processes and situations. As they show, each of them is primarily a standalone case. But in addition to the common elements described in Paragraph 6.2, it is safe to state that software generally does not seem to have been created with sustainability in mind, with the exception of the very recent eSTeP case. The outcome is discomfort with the situation: service disruption, data inaccessibility, non-functioning digital art objects, history gaps, etc. It seems that in most cases such consequences could basically have been avoided if some precaution measures had been taken from the outset.

7 International initiatives

7.1 Introduction

Although no attempt will be made to give a full account and an exhaustive overview of international activities, some elements of international programs and organisations will be presented to show that international cooperation in this domain may well be effective and worthwhile.

Part of this overview is composed from an earlier inventory made by the Alliance for Software Sustainability in the Netherlands, whose members deserve credit for it.

Wikipedia also has an entry on Digital Preservation Projects/activities³¹ with over 20 projects, which however seems far from complete.

7.2 Knowledge Exchange group (KE)

In 2005 the Knowledge Exchange (KE) group was formally established by four partners, Jisc, SURF, DFG and DEFF. In January 2013 they were joined by CSC as a fifth member. The KE group focuses on digital technologies for innovative opportunities to advance research and higher education in which open scholarship is one of the opportunities. The KE group decided in 2015 to also focus attention to software sustainability. They organised their first workshop, with more to follow.

7.3 UNESCO Persist

[<https://www.unesco.nl/digital-sustainability>]

UNESCO is concerned with cultural heritage in general. The UNESCO Persist programme focuses on software sustainability, among other topics. The UNESCO works closely together with major vendors to address digital heritage issues to come to practical solutions to legislative, copyright or patented digital material barriers.

7.4 Digital Preservation Coalition

On 1 August 2016, this UK-based organisation turned itself into an international organisation. Due to the recent nature of this decision, the DPC will be discussed under "UK" here.

7.5 Software Heritage Project

On 30 June 2016 a new international project started in France, presently referred to as the Software Heritage Project. It involves a huge effort, which emerged from INRIA with acknowledged support from significant parties such as UNESCO, IEEE Software, ACM Sig Soft, Informatics Europe, FSF, OSI and others. From the Netherlands, the NLnet foundation has donated funds to this unique project to help kick-start it. The Software Heritage mission is to collect, organise, preserve and share as much software (especially source code) for posterity as possible. This is complementary to other efforts, such as the Internet Archive, which do not have an adequate preservation strategy for these kinds of complex code resources. The project was due to go public by the end of June 2016 and has already collected 2.5 billion artefacts from tens of millions(!) of software projects so far – a collection of significant cultural, scientific and historical importance. Not only was the Software Heritage able to take over the entire Google Code archive (which was taken down with millions of projects), it has also salvaged all projects on Gitorious and a few other online repositories with large amounts of code. It also keeps track of other large live sources, such as Github and GitLab.

As of 1 July, DANS is official partner in this project, where it will provide access to the material at hand once a user interface has been developed.

³¹ https://en.wikipedia.org/wiki/List_of_digital_preservation_initiatives

7.6 PLAN-E

[<http://Plan-Europe.eu>]

The Platform of National eScience Centers in Europe (PLAN-E) was formed in 2014 by the public, mostly university-linked eScience centres in Europe that have a national or regional role. This platform is concerned with promoting eScience and data research as a research domain and with sharing and promoting the common interests of this community. Software sustainability and data stewardship are on the permanent agenda of this platform. This is all the more important because the members of this platform are supposed to have direct and concrete access to the educational channels of the universities, both for graduate and for post-graduate education.

PLAN-E membership is based on agreed Terms of Reference and the platform now has members from over 20 countries. Conceptually PLAN-E resembles the Dutch ePLAN platform.

7.7 Commercial initiatives

This is just a reminder that industry is concerned with heritage preservation, in particular digital heritage, mostly but not entirely data-oriented. Mentioning names means missing out most, but IBM, Google and EMC are worth mentioning. IBM were involved in software preservation at an early stage, Google are investing a lot in digitizing documents and creating archives (on the other hand, they show more link rot than any other party, which is not their fault) and EMC³², originally mainly involved in storage solutions and now offering complete cloud solutions and more, has a very nice page on cultural heritage initiatives³³.

7.8 UK

The UK has been active in the field of Digital Heritage and data and software curation as well as hardware conservation.

7.8.1 Centre for Digital Heritage, University of York

[<https://www.york.ac.uk/digital-heritage/>]

The Centre for Digital Heritage is an international research centre that brings together researchers from the Universities of Aarhus (Denmark), Leiden (Netherlands), Lund (Sweden), Uppsala (Sweden), Zhejiang (China) and York to undertake interdisciplinary research into digital heritage, including data management, analysis and visualisation. The Centre is open to all researchers in these exciting fields, and to their external collaborators. No special attention is dedicated to software.

7.8.2 Digital Preservation Coalition

[<http://www.dpconline.org/>]

The Digital Preservation Coalition (DPC) enables its members to deliver resilient long-term access to digital content and services, helping them to derive enduring value from digital collections and raising awareness of the attendant strategic, cultural and technological challenges they face. It achieves its aims through advocacy, workforce development, capacity-building and partnership. The DPC has a very extensive list of members, private and public, from the Bank of England and the European Central Bank to the British Library, from universities to the BBC. It also boasts an extensive list of associate partners around the world.

DPC has scheduled a publication on software sustainability in 2017.

From 1 August 2016, membership of the DPC has been open to international organisations (the NCDD was already an associated member). From now on, the DPC will therefore be considered an international organisation.

7.8.3 The Software Sustainability Institute (SSI)

[<http://www.software.ac.uk/>]

The SSI is a national facility in the UK for cultivating world-class research through software. The Institute is based at the universities of Edinburgh, Manchester, Oxford and Southampton, and draws on a team of experts with a breadth of experience in software development, project and

³² <http://www.emc.com/corporate/emc-at-glance/corporate-profile/index.htm>

³³ <http://www.emc.com/corporate/sustainability/strengthening-communities/heritage.htm>

programme management, research facilitation, and publicity and community engagement. Sustainability means that the software of today will be available – and continue to be improved and supported – in the future. The goal of the SSI is to make it easier to rely on software as a foundation of research. They help people to learn the skills needed to build better software, and ensure that people get credit and encouragement for developing sustainable software. They work with research groups from all disciplines, which gives them a broad understanding of the problems researchers face when using software. They use this expertise to ensure that the right choices are made about the software used – and pitfalls are avoided. The SSI works with researchers, developers, funders and infrastructure providers to identify the key issues and best practices surrounding scientific software.

One of the instruments the SSI uses to get their message across and develop tools and problem insights in different disciplines is the fellowship. The Institute has appointed 17 fellows to take forward their plans to improve research software and make a difference in their research domains.

The fellows represent domains ranging from archaeology to psychology and from computer science to earth systems.

How do scientists develop and use scientific software? Many researchers spend time programming new software tools for their research. Software testing is usually neglected, and sustainability is a subject that is often overlooked. According to SSI director Neil Chue Hong, sustaining the function of scientific software is more important than sustaining its form. The challenges of sustaining software involve much uncertainty surrounding authorship, boundaries, versioning and granularity.

The core of the SSI mission is to stimulate the reuse of software. This is not an easy task, because there is little reward in it. The problems are as difficult in the humanities as they are in the sciences. What the SSI started doing more than ten years ago is not the same as what it is doing today. In 2001, the UK eScience programme prepared the ground for the SSI. OMII-UK (<http://www.omii.ac.uk/>) provided and supported free, open-source software for the UK e-Research community. The project came to an end in December 2010. OMII-UK helped new users get started with e-Research, and provided software and support. The project also improved the provision of e-Research software by commissioning developers to create software to meet the needs of the research community. However, OMII appeared to be too expensive to be effective in the long run.

In a second effort, a project approach was followed. Project funding works fine to sustain software as long as you have project money.

A third endeavour concentrated on cultivating and nurturing software, for instance in the Engage programme. This worked better, and from this grew the SSI in its present form (see <http://www.software.ac.uk/>).

There is now less money available for software sustainability. EPSRC, for instance, estimated that over the last five years it has invested approximately £9m a year in software.

SSI offers expertise, not money. SSI has the policy and the tools to support community software development. The key drivers are researchers who want to be productive and at least not worse than their peers. This is their reward for sustaining their software. The approach is one of self-reliance. Researchers should be selfish about their software: they do it primarily for themselves. Better software leads to better research. Approaches relying on the altruism of researchers run a great risk of failing.

Sustainability is less about the robustness of the code than about the robustness of the team that developed the software.

The SSI now has nine staff members. Their main activities are community engagement (e.g. by providing fellowships), consultancy (such as software evaluation), policy development, and providing publicity and training.

The SSI works in cooperation with CeRCH (Centre for eResearch) at King's College London. The size of the community that wants the software to survive is crucial, which can be demonstrated by the use of the software. The software needs to be so robust that it can be installed and used by others.

Services of SSI are:

- SSI Guides
- Briefing papers
- Blogs to spread information and opinions

- The much appreciated service “Ask Steve!” (for technical questions)
- International software carpentry initiative to teach software-development to researchers
- Software sustainability surgeries: advice on code that users are developing
- The meta journal “Journal of Open Research Software”.

The SSI has 15 fellows, who act as software ambassadors in their domain. They are not developers, but researchers who lead the software development and who use software. Their costs taken together are less than one staff member (FTE).

Lessons learned:

- Providing general training on software development for research tools is not cost effective.
- It is easier and more efficient to build and sustain products built on platforms and pipelines than to make existing tools more generic.
- Setting priorities for the reuse of software is unavoidable.
- Build and sustain the right skills.
- International, large scientific collaborations and joint software efforts are useful.
- In summary: do not focus on preserving software, focus on efficiency of research, re-use, skills, and collaboration. Software sustainability is not easy, but it is possible.

7.8.4 Future-Proof Computing Group, University of Portsmouth

[<http://www.port.ac.uk/centre-for-cultural-and-industrial-technologies-research/research-areas/#future>]

The Future-Proof Computing Group at the University of Portsmouth is part of the wider Creative and Cultural Industries Faculty. One of the themes that have emerged from recent experiences is that many complex objects in software, art and archaeology are hybrid, i.e. partly physical, partly digital. This presents a new level of challenge for memory institutions that may have a different set of rules for each kind of object (such as for paintings, drawings, sculptures per material type, musical instruments, etc.).

7.8.5 The Computer Conservation Society

[<http://www.computerconservationsociety.org/>]

This UK-based society is concerned with the conservation of hardware (computers and peripherals), but also devotes attention to developing preserved software for historical computers, many of which have long since passed into oblivion. According to the CCS’s website³⁴, in order to make this activity meaningful and to allow for the possibility of writing new programs for dead computers it is sometimes necessary and always useful to implement emulators: programs which run on modern computers but which interpret programs written for the target machine and cause them to be executed in much the same way as the hardware of the original computers once interpreted the instructions of their programs and carried out those instructions. So they create emulators and keep a list of them. That also the emulators themselves get outdated can be deducted from the fact that some of them are written for MS-DOS machines! Interesting website, anyway.

7.9 Australia: Australian National Data Service (ANDS)

[<http://ands.org.au/>]

The Australian National Data Service (ANDS) was established in January 2009. ANDS was originally created as part of the National Collaborative Research Infrastructure Strategy (NCRIS) initiative to ensure that research data is used as effectively as possible by Australian researchers. The Super Science initiative announced in May 2009 provided additional funding from the Education Investment Fund (EIF) to establish the Australian Research Data Commons (ARDC). ANDS exists to transform Australia’s research data environment by making Australian research data collections more valuable through managing, connecting, enabling discovery and supporting the multiple use of this data. The purpose of this activity is to enable richer research, more accountable research, more efficient use of research data, and improved provision of data to support policy development. The outcome of this activity will be that Australia’s research data as a whole will become a nationally strategic resource. Rather than institutions just meeting ANDS’

³⁴ <http://www.computerconservationsociety.org/software/software-index.htm>

goals, the team will encourage organisations, research groups and researchers to realise their research data ambitions.

Since the end of 2013 researchers across all disciplines that use research data are represented in the Australian Research Data Commons, and nearly all research institutions have improved their research data management, leading to routine publication of their data with ANDS persistent identifiers into a data store that feeds information to the ANDS collections registry. In addition, researchers are able to find a wide variety of datasets using the ANDS data pages through a variety of discovery paths, and more institutions will be successfully engaged in meeting their responsibilities described in the Australian Code for the Responsible Conduct of Research. Most importantly, ANDS has engaged the research community to the extent that for researchers to publish their research data has become standard practice.

Suggested questions to assess software development according to ANDS:

- Does the institution have the capacity to develop or modify software as required?
- Does the institution have the expertise to recommend alternate solutions to time-intensive custom-developed development or modification?
- If a dataset is generated by custom-developed software, has that software (and the relevant documentation) been archived?
- If a dataset is generated by commercial or off-the-shelf software, have any customisation or configuration settings been archived? Are there alternative open-source tools available for other researchers without access to commercial software?
- If a dataset requires custom or off-the-shelf software for basic access (e.g. decompression), have any settings or customisations needed to access the data been specified with the metadata?

ANDS does not presently have a direct link to software sustainability.

7.10 Denmark: Royal Library

7.10.1 DEFF, the Danish Electronic Research Library

[<http://www.deff.dk/english/>]

DEFF, the Danish Electronic Research Library, is a member of the Knowledge Exchange Group and therefore involved in software sustainability research.

7.10.2 Danish Royal Library

[<http://www.kb.dk/en/nb/>]

The Royal Library maintains two collections containing electronic games; one physical and one completely digital. Through the Danish Legal Deposit Act the library acquires games published on physical media types (CD-ROM, DVD-ROM, cartridge, etc.). Presently their collection of physical video games comprises approximately 1.300 "Danish" games for a wide range of game systems and OSs. The other collection is the Web archive Netarkivet which is maintained by The Royal Library and The State and University Library in Aarhus. They perform extensive harvests of Danish websites including online game publications. Presently they have at least 120 functioning (Danish made) online games in their Web archive and probably an even larger number of defunct or partially defunct games. The games are in various formats including Unity3d, Flash, and Java. The Library also is in possession of a small number of "download only" console games (WiiWare, Xbox Live Arcade, etc.) that are currently only stored on the consoles themselves. No app games (iOS, Android, etc.) are in the collection yet. (Taken from the website.)

7.11 France

Bibliothèque Nationale de France (BNF) has a Web archiving team which is, among other things, working on Flash, Flash video and streaming content.

The French academic research landscape is scattered³⁵: in terms of national budget there are 20 main research institutes, among which CNRS (19%), CEA (14%), INRA (5%), Inserm (5%) and Inria (<2%). There is presently no software-specific policy in place and the academic software

³⁵ The following is based on information provided by the French membership of PLAN-E (Dublin, 9-10 May 2016)

producers are free to organise their software production and diffusion. However, there are a few initiatives regarding software sustainability with a national scope: SourceSup, CeCILL, Plume, and some professional networks and entities to link academic work to industry and SME's.

- SourceSup³⁶: national academic forge "La forge Enseignement supérieur et Recherche", provided by CRU and after that by Renater (NREN) since 2004. This project hosts software development projects for academic research and universities and their collaborations. Currently there are 489 visible hosted projects, all open source technologies (subversion, git, Jenkins, Sonar, Nexus...). Authentication is done with the "federation Éducation-Recherche" identity federation (266 identity providers and catch all for external collaborations). It has a complete set of functionalities: link with Renater mailing lists server (Sympa³⁷), possibility of project website, building, deploying and automating tools (Jenkins), code quality (Sonar), documentation management (Nuxeo). There are other institutional or thematic forges.
- CeCILL licenses³⁸. The CeCILL family licenses are developed by CEA, CNRS and Inria since 2005. The arguments: "Today Free Software is important in the scientific community as well as in administrations and in the enterprise. Nevertheless, the use of licenses created in the US, such as the GNU General Public License raises some legal issues. These issues may lead to uncertainties that may prevent some companies and organisations to contribute Free Software. To provide a better legal safety while keeping the spirit of these licenses, three French public research organisations, CEA, CNRS and Inria launched a project to write Free Software licenses conforming to French law. " CeCILL-A is designed to be fully compatible with GNU GPL.
- Plume³⁹: to Promote economical, Useful and Maintained software For the Higher Education And THE Research communities. Plume is mainly supported by CNRS 2006-2013. The portal provides a software catalogue, mainly Free/Libre Open Source Software (FLOSS) that is used and/or produced in universities and national research organisations, laboratories or departments. It presently involves 2200 members, 950 contributors, 18 themes, 1270 index cards, several keywords levels including institutes and user's classification, and is based on open source technologies (Drupal...). Typical actions: topical workshops and training sessions oriented to software development (tools, dissemination, IPR and licenses...) or targeting community building around software interest, cross-disciplines experience sharing. Due to lack of resources the project was frozen in mid-2013, but the information and the platform are still online.

Plume & research software: Several research units (ICJ, LAAS, LIGM) required research software descriptions in Plume to disseminate their software production. Some results: 350 research software products produced in French laboratories descriptions (1/3 translated in English); actions on software dissemination (documentation, guidelines, about 10 workshops); work on legal issues (French IPR laws and licenses); PLUME-FEATHER: <https://www.projet-plume.org/eng>.

Professional networks:

- DevLOG⁴⁰: software developers, officially supported by CNRS since 2001. Mailing list: 1000 subscriber. Main actions: JDEV annual conference, topical workshops and training sessions (software development, dissemination, IPR and licenses), community building, cross-disciplines experience sharing
- RBDD⁴¹: databases professional network officially supported by CNRS since 2012. Mailing list: 350 subscribers. Main actions: recommendations, workshops and training sessions (databases and related tools development, dissemination, IPR and licenses, cross-disciplines experience sharing).

Other professional networks are focused on regions, disciplines (scientific computation such as "Groupe Calcul"...) or institutes. They are federated at the national level.

Future initiative: IdGC together with other CNRS laboratories and European partners have submitted a proposal about software, awaiting granting.

³⁶ <https://sourcesup.renater.fr/>

³⁷ Sympa is an open source mailing lists server created by CRU and provided as a service by CRU then Renater since 1997 for the research and universities members

³⁸ <http://www.cecill.info/index.en.html>

³⁹ <https://www.projet-plume.org>

⁴⁰ <http://devlog.cnrs.fr/>

⁴¹ <http://rbdd.cnrs.fr/>

7.12 Germany

The DFG is a member of the Knowledge Exchange Group.

7.12.1 Technical University of Munich

1) [<https://sustainability.wiki.tum.de/Sustainable+Software>]

The Technische Universität München has attention for sustainable software rather than for software sustainability at large. The emphasis is on designing software that not only can be maintained but is also environment-friendly (green software).

2) [<http://www4.in.tum.de>]

The Fakultät für Informatik, dep. „Software and systems Engineering Research Group“, does research and provides services on software maintenance. It has a competence centre for software maintenance (<http://www4.in.tum.de/research/maintenance/index.shtml>) and a competence centre for the assessment of software quality.

3) A recent publication by Joan Morales Tua presented at the university, “Metrics for Sustainability and applicability to Software” addresses sustainability criteria to software (http://www4.in.tum.de/lehre/seminare/WS1112/UPC-EnviroSiSE/Joan_KPIforES.pdf), in the context of environmental sciences, but not specifically the sustainability of the software itself.

7.12.2 University of Freiburg

[http://bw-fla.uni-freiburg.de/wp-uploads/bw-fla.uni-freiburg.de/2012/03/strategy_paper_KEEP_expert_workshop_Joining_Forces_Berlin_final.pdf]

The University of Freiburg has a digital preservation group at the computer science department working on different aspects of emulation, such as emulator integration (offering, among other things, emulation to library and archive users), emulator testing (automatic testing of new versions, regression tests, unit testing, workflows to integrate new versions of emulators before using them), measurement and metrics for runtime, extended object life-cycle management. They design systems that can be virtualised and later emulated to ensure accessibility after the out-of-service declaration, workflow automation of any interactive GUI digital ecosystems (independent of operating systems such as Windows, Mac OSX or Linux).

The group is also working on software archiving on different levels: from archiving the single software components to complete installations of original environments and complete machine dumps.

7.12.3 Karlsruhe University

Another publication, realised by the Forschungszentrum Informatik of the University of Karlsruhe, is called Towards Software Sustainability Guidelines For Long-living Industrial Systems, by Heiko Koziol, Roland Weiss, Zoya Durdik, Johannes Stammel, Klaus Krogmann, Industrial Software Systems, ABB Corporate Research Ladenburg, Germany, Forschungszentrum Informatik (FZI), Karlsruhe, Germany (<http://subs.emis.de/LNI/Proceedings/Proceedings184/47.pdf>).

7.13 The Netherlands

7.13.1 SURF

[www.surf.nl]

SURF is the collaborative IT organisation for Dutch higher education and research.

Among many other things SURF runs the Coordination Office for Research Data Management including, for the time being, software sustainability. This activity is allocated to SURFsara. SURF is also a member of the Knowledge Exchange group (but leaves the practical representation to NLeSC and DANS).

7.13.2 NWO

[www.nwo.nl]

NWO is the Netherlands Organisation for Scientific Research (the national funding organisation). It has funded CATCH (Continuous Access To Cultural Heritage), one of the projects in the use case list (see 6.4). In that sense NWO has demonstrated the importance of actions towards the sustainability of cultural heritage. However, the software developed under that programme was

not yet constructed with sustainability in mind in the practical sense, as can be read in the use case. NWO, however, is involved in the SURF coordination efforts for research data management and is preparing action towards sustainability of data and software resulting from funded projects.

7.13.3 DANS

[www.dans.knaw.nl]

DANS (Data Archiving and Networked Services) is an institute founded by NWO and the KNAW (Royal Academy for the Arts and Sciences). DANS is concerned with providing data services to researchers. DANS provides various sorts of data storage, archiving and access services, adds value to datasets and archives by adding metadata and making them searchable. DANS hosts no storage facilities. Together with NLeSC, DANS is responsible for putting software sustainability on the map and on the national and European agendas. Although SURF is the KE member from the Netherlands, in practice DANS (with NLeSC) fills in this membership regarding software sustainability, both policy-wise and technical. Together, DANS and NLeSC have issued the vision document Research Software at the Heart of Discovery, which deals solely with software sustainability. DANS has also published a policy document directed to the practical implementation of policies: A Conceptual Approach to Data Stewardship and Software Sustainability. Moreover, DANS has conceived the idea of a Software Seal of Approval to be allocated to software creators and/or software products that have taken into account prescribed conditions. This idea is being implemented internationally, together with the SSI (UK) and the KE for starters.

7.13.4 NCDD and NDE

[<http://www.den.nl/pagina/511/netwerk-digitaal-erfgoed/>]

[<http://www.ncdd.nl/>]

The *Nationale Coalitie Digitale Duurzaamheid* (NCDD) (National Coalition for Digital Sustainability), part of the *Netwerk Digitaal Erfgoed* (NDE) (Network for Digital Heritage), has started research activities towards software sustainability for the Cultural Heritage sectors, the National Archive, the National Library, and the Sound and Vision and Science domains. The Science domain is represented by DANS, to which the compilation of this document has been allocated. This document is a result of that activity.

7.13.5 NLeSC

[<http://www.esciencecenter.nl>]

The Netherlands eScience Center (NLeSC) is a public organisation with a national scope, concerned with accelerating scientific discovery, through the supported use of the most modern IT and innovative ways to deploy the IT-infrastructure as a whole in scientific cooperative projects. Together with DANS, NLeSC contributes to national policies regarding data stewardship and software sustainability. Policy documents are issued and presented to policy and funding organisations in the Netherlands. NLeSC has defined an eScience Technology Platform (eSTeP – see Paragraph 6.7), covering the internal research within NLeSC to sustain the quality of its services together with a catalogue of quality approved software (either developed at NLeSC or adopted from elsewhere but with proven quality). NLeSC is concerned with re-usability of software and data and with creating software with a minimal maintenance footprint.

7.13.6 ePLAN

[<http://escience-platform.nl>]

The Platform of eScience/Data Research Centers in the Netherlands (ePLAN) is formed by the public, mostly university-linked eScience and Data Science/Data Research centres in the Netherlands. This platform is concerned with promoting eScience and data research as a research domain and with sharing and promoting the common interests of this community. Software sustainability and data stewardship are on the permanent agenda of this platform. This is all the more important because the platform members are supposed to have direct and concrete access to the educational channels of the universities for both graduate and post-graduate education.

7.13.7 Legacy Coalitie NL

[<https://www.dutchdigitaldelta.nl/en/about-us>]

[<http://www.nwo.nl/over-nwo/organisatie/nwo-onderdelen/ew/bijeenkomsten/legacy+coalitie+nl>]
(only in Dutch)

The Dutch Digital Delta, a cooperative organisation supported by NWO, EZ, TNO and ECP, has launched an initiative under the name of Legacy Coalitie NL. The goal of this effort is to improve the management of the negative effects of software legacy (in order to improve the competitiveness of Dutch business and industry). This initiative, the contents of which are led by NWO and Inseit (<http://www.inseit.nl/>) started with a kick-off meeting on 12 January 2016. The idea behind this initiative is that a large amount of IT expenditure, (75% is mentioned) goes to maintaining legacy software. Improving this situation should yield a competitive advantage. An activity like this touches on the domain of software sustainability, but from quite a different point of view and not aimed at maintaining heritage. Rather the opposite. From the outcome of the research by this coalition, however, a lot may be learned regarding proper coding practices for building maintainable software. This adds to the eSkills that should be part of education and training schemes for making software in the first place.

7.14 USA

7.14.1 CSESSP Workshop

[<https://www.orau.gov/csessp2015/agenda.htm>]

The Computational Science and Engineering Software Sustainability and Productivity Challenges (CSESSP) Workshop (Hilton Washington DC/Rockville Hotel & Executive Meeting Center, Rockville, MD, 15-16 October 2015) yielded quite a lot of relevant presentations, which can be obtained via the Web link above.

7.14.2 Software Infrastructure for Sustained Innovation (SI² - SSE&SSI)

Software is an integral enabler of computation, experiment and theory and a primary modality for realizing the Cyberinfrastructure Framework for 21st Century Science and Engineering (CIF21) vision, as described in <http://www.nsf.gov/pubs/2010/nsf10015/nsf10015.jsp>. Scientific discovery and innovation are advancing along fundamentally new pathways opened by development of increasingly sophisticated software. Software is also directly responsible for increased scientific productivity and significant enhancement of researchers' capabilities. In order to nurture, accelerate and sustain this critical mode of scientific progress, NSF has established the Software Infrastructure for Sustained Innovation (SI²) programme, with the overarching goal of transforming innovations in research and education into sustained software resources that are an integral part of the cyberinfrastructure.

SI² is a long-term investment focusing on catalysing new thinking, paradigms, and practices in developing and using software to understand natural, human, and engineered systems. SI²'s intent is to foster a pervasive cyberinfrastructure to help researchers address problems of unprecedented scale, complexity, resolution, and accuracy by integrating computation, data, networking, observations and experiments in novel ways. NSF expects that its SI² investment will result in robust, reliable, usable and sustainable software infrastructure that is critical to achieving the CIF21 vision and will transform science and engineering while contributing to the education of next generation researchers and creators of future cyberinfrastructure. Education at all levels will play an important role in integrating such a dynamic cyberinfrastructure into the fabric of how science and engineering is performed.

It is expected that SI² will generate and nurture the interdisciplinary processes required to support the entire software lifecycle, and will successfully integrate software development and support with innovation and research. Furthermore, it will result in the development of sustainable software communities that transcend scientific and geographical boundaries. SI² envisions vibrant partnerships among academia, government laboratories and industry, including international entities, for the development and stewardship of a sustainable software infrastructure that can enhance productivity and accelerate innovation in science and engineering. The goal of the SI² programme is to create a software ecosystem that includes all levels of the software stack and scales from individual or small groups of software innovators to large hubs of software excellence. The programme addresses all aspects of cyberinfrastructure, from embedded sensor systems and instruments, to desktops and high-end data and computing systems, to major instruments and facilities. Furthermore, it recognises that integrated education activities will play a key role in sustaining the cyberinfrastructure over time and in developing a workforce capable of fully realizing its potential in transforming science and engineering.

8 Software types

8.1 Semantics

Various terms are being used to refer to software, of which software, (computer) program, tool, application, app, code, source, interface and firmware seem to be the most common. We will not try to produce strong definitions, as these terms have grown their own domains of deployment over decades and are commonly used quite synonymously to a large extent. Yet their differences are characteristic as well. As an introduction to a more detailed classification of software, a shorter and more intuitive approach to characterizing these common terms is given below. Software will be defined as the generic term denoting any piece of readable computer code in any type of language including assembler and microcode, from which by an interpreter or compiler an executable version can be made to be executed on a specific or general computer system. It is to be noted that in some cases⁴² a distinction has been made between software and application in that software were to be synonymous to system software in contrast with an application. An application is then a code that uses system software to run on a computer. We do not follow that distinction.

Under system software a whole new category of distinct functionalities can be discerned, such as the operating system, the runtime system, the compilers, the scheduler and more.

A (computer) program, although basically the same as software, will be used more as a synonym to application.

A tool is a piece of software that serves a specific mostly technical function on a computer or in a computer program. So there are tools to analyse data, to apply specific changes to photos, to tune the behaviour of a program or a computer, to convert formats or data or file types into each other, and other comparable actions. Specifically, linguists' tools encompass parsers, tokenisers, taggers, named entity recognisers.

An application is a piece of software that serves a specific (end-user) goal to conduct research, process and analyse data, up to and including running major processes, such as in banking, insurance companies' transactions, traffic control and other monitoring and basically all other computer activities not named differently. It can also be considered a piece of software that is shipped as a unit, can be installed (using installers) on a computer system, and – when properly created – has a history of versions and updates. Enterprise applications can be very extensive in nature.

An app is a relatively new phenomenon that emerged with the smartphones. It is basically a pre-compiled piece of software (mostly of the "tool" type), running on a specific type of hardware or even only under a specific version of the operating system to perform a specific and mostly rather limited number of functions. But games and other types of programs are also called apps if designed to run on a smartphone or mobile device. Although it is the purpose of this exposé to try to distinguish between the terms mentioned, in actual fact there is a continuum from website to app to application.

The term code refers to the very content of a piece of software, either in original readable form (source or source code) or in machine readable format (machine code). The physical contents of all other terms can be referred to as code. Running Java involves Java producing readable instruction code (comparable to an assembler) for a virtual machine. This code is called "byte code".

A source is any piece of software that is still in readable form (in its "original" state) in contrast to the end stage before the software can run on a computer, namely the executable code.

An interface is a piece of software that communicates with two (or more) different environments. Such an environment can be different programs, operating systems, devices or also a device (computer) and a user (user interface).

Firmware is software installed by "the firm" that produced the hardware and comes with the device (a CPU, memory module, disc drive, USB device, ...). Often, it is not changed after installation, but increasingly more often it is updated.

8.2 Software classification

In order to get a better view of what it takes to certify software, an attempt is made to distinguish different types of software. The way in which this is done is by no means unique, but below different axes of characteristics are presented, which provide a multidimensional picture of the matter. After that the domains from this multidimensional space can be selected that could be

⁴² This distinction was introduced by the US partners in the EXASCALE PROJECT and was not agreed upon

addressed by a seal of approval. This can subsequently be used to narrow down a particular discussion on software sustainability in a domain to a limited set of software types.

8.2.1 Firmware

Firmware is software of which the executable runs at the heart of a device: the disc drive, the network interface, the CPU, the USB-device and many other small devices such as used in automobiles. It is essential software to make the hardware function as it should. It resides on the hardware component as an essential part of it and is as such not part of the operating system. The term is presently eroding because in the world of smartphones the Operating System is often also denoted as firmware. And Operating Systems generally interact closely with the firmware of the peripherals (disc, memory, sound and video cards, screen, USB ports etc.).

Firmware used to be rarely adapted after installation, but increasingly more often is needed to be changed. Changes may introduce additional functionality or improvement of the behaviour of the device, but may also be necessary to adapt to changing environments (the OS of the most machine for example).

For the sake of this document a mobile device's operating system will not be called firmware but be classified as an operating system.

In the context of software sustainability firmware may be relevant when emulating the environment in which obsoleted software once ran.

8.2.2 Embedded software

Embedded software is quite comparable with firmware. The term is mostly used for non-computer and even non-IT types of equipment. Such as washing machines, Microwave ovens, DVD-players, TV-equipment, but also in automobiles, robots and manufacturing machines.

8.2.3 System software

System software refers to the software environment that enables a (computer) system to function and interact at whatever minimum level with a user. User in this case may be a system administrator or end user depending on the device and/or on experience and/or organisational circumstances.

System software may be written in assembler code or in higher computer languages or may consist of a mixture of both. It consists of a suite of programs that together provide the functionality a computer system can deliver to its user(s), individually or collectively, and makes the computer system generally usable. System software is largely dependent on the composition of the computer in terms of processors (one or more CPUs, GPUs, FPGAs, etc.), their interconnect(s), the memory and memory hierarchy, the connections to the world outside the computer (network, external devices) and the services the computer is to provide.

8.2.3.1 Operating system

Standalone computer systems normally have a single operating system (OS). Through emulation other operating systems may be enabled to run on the leading operating system. The operating system provides most of the characteristics that the system software provides to the user.

These days, virtualisation has entered the arena. Through virtualisation users may bring their own operating system on to a system in a multi-user environment. This means that the basic operating system is a thin layer on top of which a variety of operating systems, up to and including homegrown operating systems can be temporarily installed. This type of virtualisation comes on top of other types of virtual machines or devices, such as a virtual disc spanning several physical disks (could be RAID technology).

Examples of today's operating systems are Microsoft's Windows-n, Apple's OS-n and IOSx, Unix, and Linux.

8.2.3.2 Drivers

Because computers are composed of different individual components, drivers are required to interact to and from these components with the operating system. Components are the CPU or other processing units, the memory, the disks, the video and audio devices, the various ports (USB, VGA, HDMI, Ethernet, phone, etc.) and the equipment attached to these ports, the mouse, touchpads, touchscreen, and more. Drivers are sufficiently independent from the OS so they can be individually updated when the devices change, to improve functionality, correct errors or when the operating system (version) changes. Drivers can often be installed on the fly when new equipment is attached to any of the ports, but sometimes the operating system requires to be rebooted in order to correctly address ("recognise") the new drivers.

8.2.3.3 Networking software

To a certain extent networking software acts as a driver to an operating system. But the interwovenness of the general purpose OS with the networking environment is often more complex.

Another reason to distinguish networking software is the existence of so called routers. Routers are computers dedicated to handle the routing of data to and from their origins and their destinations. In such computers the networking software, the different protocols that may exist in handling data in their various formats is the dominant component.

Networking software may be divided in software handling the communication with the environment external to the computer ("the Internet" for example) and the environment that is internal to the computer. The distinction is only partly valid, as most larger computer systems are composed by a cluster of individual systems through a dedicated network, of which the complexity may be to the extent that such a cluster behaves as a single computer to the user. As most computers today have more than one processor on a chip and more than one chip, most computers have an internal network to do the communication between the processors and the physically distributed memories.

8.2.3.4 Compilers and interpreters, runtime systems

For a computer to actually do something for a user the user must be able to tell the computer how to use its resources to come to a result. The way to do that is by writing a computer program, however small or large. This can be done by writing in assembler, in which case one writes the code more or less at the level of hardware instructions, or using any type of higher computer language⁴³. The higher computer language needs then to be translated into hardware instructions. If the computer language allows the written code as a whole to translated into instructions one says the code is "compiled" and the compiler is a piece of computer code that does just that. The compiled code is then called an executable. The executable requires a "runtime system" that provides for various links between the executable and operating system. Some languages only are supported by an interpreter. The interpreter is a piece of computer code that translates the higher level language lines one by one. So no separate executable is produced first.

Examples of languages that use compilers are Fortran, C, C++. Examples of languages that are interpreted or compile to byte code are Java, Python, Shell, PHP. Also scripting/mark-up languages are interpreted. Examples are HTML, XML, etc.

Strictly speaking compilers do not need to be part of the system software, although the runtime system may be required to part of the system software, either for the privileges that may be required to access parts of the system and/or because the runtime system and the compilers are to be shared by all users of a system. If the runtime system only uses system features that the operating system allows all to use, user can install private versions of compilers (such as a Fortran compiler under MS Windows or Apple OS).

8.2.3.5 Script

The term script is ambiguous – even within the context of computers – and therefore difficult to position in this overview. There will be a second item for scripts under "Internet software". By also placing it here under system software, another type of scrips can be described: a man-readable set of instructions directing the runtime system to perform a series of tasks, semi-automated, or repeatable. The scripting language has to have a prescribed format and the runtime system must have the feature to support/understand the script being offered for execution. Scripts are also used to establish the environment in which a user-program works, where the files are or have to go, what pre- or post-processing needs to be done etc.

A script could also be defined as a user program in its own right, but because it will usually involve formal OS-type tasks rather than computation, it is for the sake of this overview placed under system software.

8.2.3.6 Libraries

Libraries are collections of frequently required programs or sub-programs, provided ready to use to programmers and developers. Libraries contain pieces of code that can be re-used, often in optimised form. It helps users not having to redo what has been done already and most likely better than individual users can, except for very experts. This may concern pieces of scientific routines, statistical routines up to codes for Web design or visualisation. Because libraries are often provided to all users in an environment, the libraries may be considered part of the system software, but the same considerations apply to libraries as to the compilers and interpreters.

⁴³ We will not dive into the very formal definitions of what defines a computer language, compared to other code generating facilities that have all the characteristics of a language but are formally not accepted as such. Fortran, the most widely distributed tool for writing computer programs in the scientific computational domain, being one of those examples.

8.2.4 Application software

Application software is generally what end users write or use to get a computer do what the end user wants to be done by the computer. Whereas the number of operating systems, languages and compilers is large but countable, the number of applications is numerous and growing. One can still distinguish applications after their nature. Application software uses system software and the hardware to run.

Applications in science, but also in other environments, may or may not be implementations of an algorithm. The algorithm itself is not considered "software", but rather a mathematical concept.

8.2.4.1 Administrative applications

Computers have been used for administrative applications from an early stage. Such applications encompass financial administrations (banking, insurance, general business), spread sheets, and perhaps also statistical applications. Also packages such as MS Office or Open Office can be considered administrative applications, however broadly used by all. Many early administrative applications have been written in COBOL, a special computer language for such purposes, and are still in use. Applications in this domain are dominantly written by commercial software companies, although typically in the database technology domain innovations are being brought in from scientific research.

8.2.4.2 Business applications

Early customer relation applications (large companies and SME's), trade market (trending) and CAD/CAM applications are examples of business applications. Due to today's exponential growth in data and the interest in data, many business intelligence applications are being developed and used. They concern data analysis in all its aspects, both structured (databases, spreadsheets) and unstructured data (Facebook, Twitter, email). The market for business software was rather limited until five to ten years ago business intelligence and data analysis became booming and many companies are trying to cover parts of that market. For security intelligence, too, many new methodologies are being developed and brought to market as general purpose tools. Applications in this domain are also predominantly written by commercial software companies, although in the data analysis domain new technologies are being brought in from the science domain.

8.2.4.3 Educational applications

This application type encompasses the many resources often distributed as CD-ROM or CD-I applications, that have been used for educational purposes (at schools, as supplements to textbooks), but also for gaming or administration (tax forms etc.). Today, a lot of education is conducted through MOOCs (Massive Open Online Courses) and Blackboard (course administration).

8.2.4.4 Scientific applications

Security intelligence, scientific research and physical construction design have been the main drivers for hardware and software development from the onset of digital and pre-digital (analogue) computers. It is the domain of scientific software applications that is presently raising and addressing the software sustainability issue.

Scientific software is predominantly written by individual users or user groups. But there are also commercial parties operating in this domain. For example for chemistry (Gaussian), Environmental Impact Assessments, Scientific Libraries (NAG), and so on. Even within the science domain a distinction is often made between application software (software used and often developed in all science domains except for computer science) and software developed in computer science.

8.2.4.5 Visualisation software

Visualisation software is developed by both research groups and commercial companies. The software focusses on the representation of data on computer displays and other kinds of screens. New developments concern typically 3D perceptions and the representations of enormous quantities of data for analytical purposes or for communication (advertisement).

8.2.4.6 Games

The development of and interest in games have driven the directions of special-purpose hardware in support of these developments. Today GPUs, Graphical Processor Units, the direct result of the requirements of the gaming industry, are being brought back to the non-graphics application domain in the form of computational accelerators. Games are basically the same in nature as other software applications, but special attention is paid to both the graphical design of the output and the attractiveness of the user interface. The same techniques are used in animation. Some games are designed specifically to run on specially designed hardware (for example Nintendo, Xbox, PlayStation).

8.2.4.7 Miscellaneous

This is just to note that this list is most likely incomplete.

8.2.5 Internet applications

The Internet allows for fast exchanges of data in all shapes and formats. Early typical Internet applications are mailers, which send rather strictly formatted messages from sender to addressee. But the “killer app” for the Internet was the software that allowed for the World Wide Web, the so called browsers. Browsers developed after the introduction of HTML (Hyper Text Mark-up Language). These browsers completely superseded earlier versions called Gopher, because HTML allowed for the exchange of non-ASCII documents, such as pictures and other graphics, sound and more.

8.2.5.1 Web applications

Web applications refer to those Internet applications that function within the scope of the World Wide Web (rather than on the Internet at large).

8.2.5.1.1 Browsers

The most important Web applications are the browsers. The browsers are software with a communication aspect, a representation aspect, an interface aspect and many handles that can be used by third parties to add functionality. They form the main vehicle for the representation of Websites and the execution of special tasks within these Websites. Browsers allow users to represent and share with the rest of the Web users their own websites, homegrown or commercially developed, increasing often using predefined templates with lots of variation options in design.

Codes that use browsers are written in HTML-n or XML and are becoming increasingly complex. They require libraries for special functionality and can even be generated on-the-fly. This makes them extremely volatile. Other tools for developing browser interpretable code are for example JavaScript. JavaScript in turn is also used in games development and other products, like PDF⁴⁴.

8.2.5.1.2 Browser tools/add-ons

Browsers generally allow for third party additions of functionality of the browsing capabilities. Such additions are often called add-ons and often have the nature of a tool. Examples are add-ons for interpreting graphics formats, including video, functionality for downloading files, links to third party applications (Adobe pdf, MS Office), security functions, special user interfaces, etc. Such tools or add-ons were often written in Java, Perl, Python, etc., but today Microsoft Active Server Pages is sometimes used and the rest is by JavaScript.

8.2.5.1.3 Server side browser tools

On the server side, it is more differentiated. Perl is still there, PHP is still strong, Python is a niche here, and Javascript is sneaking in at the server side (Node.js)!

8.2.5.1.4 Portals

Portals are basically just Web sites, but they have a special function as an entry point to a special service, such as for accessing archives, social communities or a particular type of resources.

8.2.5.1.5 Scripts

Within the context of Internet applications, a script is software to enable “dynamic” Web content. This can be done at the browser side and at the server side. Functions that have to do with security and data access are typically done at the server side, functions that support the interactions between user and webpage are typically done in the browser. Nowadays, every browser is also a scripting engine. It has been a long and winding road before the major browsers supported the same scripting language with the same semantics. The standardisation of Javascript as ECMA script in successive stages has done a lot to remedy this. With the advent of the Google Chrome browser in 2008, the performance of Javascript in the browser has improved dramatically, and all major browsers have since then upgraded their Javascript engines to a comparable performance. Both of these factors, standardisation and performance have two consequences:

1. Javascript is well underway to be the most universally supported programming language (all desktops, tablets and phones).
2. Client side scripting has become much more prominent: from accommodating mouse clicks to complete applications with extensive architecture and elaborate business logic.

Despite the fact that Javascript is on the rise now, it might well be that there will be a return to ordinary programming languages for coding the business logic of a website client side. For this it is needed that browsers implement a standardised bytecode, to which programming languages can be compiled. An attempt into this direction is already underway and is called WebAssembly (<https://en.wikipedia.org/wiki/WebAssembly>).

⁴⁴ <https://en.wikipedia.org/wiki/JavaScript>

8.2.5.2 Mail applications

Mailing (or emailing) applications may be independent Internet applications or be embedded in a browser. The concept of email is older than that of the World Wide Web, but HTML explicitly knows "MAIL" as a type. Basically email is used to send (man-readable) messages over the Internet, just like normal letters (now referred to as "snail mail"). Email is becoming increasingly more sophisticated and is not limited any more to ascii-characters, but may include the whole HTML arsenal of features, including graphics, images, links, etc.

8.2.5.3 (Data) transfer protocols

The Internet protocols are pieces of software that reside on the computers that route the Internet traffic as well as on the resources that have been linked to the Internet. Data transfer protocols are the most basic vehicle that makes the Internet work. They encompass the data that have to be transferred from A to B and describe the nature of the data. The most common basic Internet protocol is TCP/IP (Transmission Control Protocol/Inter Protocol) and UDP (for example for telephony). Applications on top of this basic layer are for example FTP.

8.2.5.4 Malware/viruses

Malware and viruses are pieces of software that one way or another misuse the gaps in the security of both the Internet and the systems linked onto the Internet (computers, other types of devices, up to household equipment). The Internet and laptop type of operating systems while not being designed with their present use in mind have many exploitable leaks and security gaps that malware and viruses are using to intrude or destroy data and/or systems and/or communication. Examples of such exploits are categorised as spammers, port scanners, denial of service attacks, ransomware, key loggers or honeypots. Also social engineering may pose a threat, but that is not of a machine intrusive nature.

8.2.5.5 Service

This term falls somewhat outside the scope of this document, because a service by itself is not a software product. And in as far as it is, it is already described elsewhere in this document. The term service is mostly related to background processes where communication plays a role, such as a process that handles printing, or communication to an environment outside the computer system. But it is also used to denote the interaction between programs, or as part of other terms such as Internet Service Provider (ISP), Software as a Service (SaaS) and similar expressions.

8.2.6 Mobile apps

Apps for mobile devices (smart cell phones, laptops, tablets, etc.) are applications with a mostly limited but precisely defined purpose. Many apps could as well have been browser add-ons or are not far from being webpages for a single Web domain. But smartphone games are also referred to as apps. Some apps make use of hardware components for very different purposes than those for which they were on board the device in the first place. Examples include positioning, 3D acceleration and horizontal levelling.

8.3 Other characteristics axes

Software can also be distinguished along other axes:

8.3.1 Open software versus closed (proprietary) software

Open software or open source software is software of which at least the source code is available for inspection, but of which regularly the creator(s) permit others to read, modify, re-use and improve the source code, provided the result remains open. Proprietary software is mostly of a commercial nature and has to be taken as is, without the right to inspect the source code.

8.3.2 Commercial software versus freeware

Commercial software can be obtained from companies on the basis of payable licenses. Freeware can be downloaded from the Internet at no cost. Licenses may be attached, however, limiting the use or requiring benefits for the provider in return. Often a payable version of the freeware exists as well, allowing the user more options and capabilities.

8.3.3 Licensed software versus unlicensed freeware

Software may be distributed with an explicit license attached, but occasionally also without one. In those cases the license is implicit (the original developer's copyrights may still exist). The original rights to the software remain in place, even if the software has become obsolete or is no longer executable due to hardware no longer being available or the operating system (version) no longer being supported.

8.3.4 Off-the-shelf software versus tailor-made software

Business applications have been delivered to companies as off-the-shelf copies. Often such programs have been adopted to the specific needs of the client, using options available in the software or with minor additions.

On the other end of that spectrum is software that has been designed from scratch, (re-)using existing tools from libraries or not. Such codes are most difficult to keep alive after major (operating) system upgrades or hardware refreshes.

8.3.5 Individual programs versus software packages

Programs often are standalone tools, but they may also be part of a software suite of related or interrelated programs. Microsoft Office and Open Office are examples of related programs. They can be used as standalone programs, but they share common parts (such as the help functions) and exchange of data is often easier within the suite. But in other situations a suite is really a suite: programs rely on other programs producing their input first before they can run. Sometimes libraries of (sub-)routines are also called packages.

8.3.6 Owned software versus software as a service

These days a transition can be witnessed from program ownership towards program sharing. This is one of the consequences of cloud developments. The introduction of Office 365 by Microsoft is a typical example. Rather than installing an owned copy of the Office software on a laptop, one shares a Web supported version that resides on a Microsoft resource. In cloud terminology this is called "Software as a Service" (SAAS).

9 Appendix

Specificaties	AUTONOMATA-POI-demo	AUTONOMATA-transcriptietoolset	COREA-coreferentieservice	DAISY-demonstrator	Dupira	Frog	Spelspiek
Versie	1.1.1	6.1	1.0	1	1.0	0.12.19	1.0
Jaar	2011	2011	2008	2012	2010-2012	2014	2007
Financier	NTU STEVIN	NTU STEVIN	NTU STEVIN	NTU Stevin	Radboud Universiteit Nijmegen, Phasar	ILK UVT	NTU STEVIN flankerende activiteiten
Eigenaar	Nuance Communications Inc.	Nederlandse Taalunie	Nederlandse Taalunie	NTU Stevin	Cornelis H.A. Koster	ILK UVT	INL, Van Dale, Selfservice Company, Knowledge Concepts
Opdrachtgever	20.1 MB	NTU STEVIN	N.v.t.	48M (gecomprimeerd)	N.v.t.	Linux	Windows
Bestandsgrootte	exe	11.2 MB	Linux	N.v.t.	Windows 7, Linux, Debian	TIMBL, MBT	Windows
Besturingssysteem	Windows	Linux, DOS, Windows	Linux, MBT	php, java, python, javascript	gecompileerde Dupira grammatica (dupira.aob en dupira.bk), zonder de broncode	Nederlands	Nederlands
Talen	Nederlands	Nederlands (Nederland en Vlaanderen), Duits, Engels	N.v.t.	Nederlands	Nederlands	Nederlands	Nederlands
Documentatie	UREC2010-artikel	LangTech2008-artikel, Interspeech2007-artikel, LREC-2006-artikel	UREC2008-artikel	N.v.t.	dupira.pdf	https://language-machines.github.io/fr/	https://tst-centrale.org/nl/diensten/nieuws/nieuwsberichten/170-spelspiek
Project	AUTONOMATA Too	AUTONOMATA Too	COREA	DAISY	Phasar/TW41P	Corpus Gesproken Nederlands	N.v.t.
Projectwebsite	http://speech.ils.ugent.be/project/autonomata-too	http://speech.ils.ugent.be/project/autonomata-too	http://www.cnts.ua.ac.be/fris/corea.html	http://hcr.cts.kuleuven.be/projects.php?project=5172	http://www.phasar.cs.ru.nl/	Corpus Gesproken Nederlands	N.v.t.
Toepassing	Dit is een demo van een service die automatisch spraak naar tekst herkent voor de gebruiker (Points of Interest).	Deze tool kan men toepassen in een tekst-naar-spraakstelsel (spreekende computer).	Automatische interpretatie van teksten.	question answering systemen	Dependency parsing voor het Nederlands; Information retrieval	Automatisch taggen en ontleden van teksten.	Online hulp bij spelling
Bijzonderheden	Beschikbaar via een evaluatie licentie van onbepaalde duur.	Voor commercieel gebruik geldt een royalty model. Neem voor meer informatie contact op met de servicedesk.	N.v.t.	N.v.t.	Maakt gebruik van AGFL	N.v.t.	N.v.t.
Categorie	Corpus	Corpus	N.v.t.	N.v.t.	Corpus	Corpus	Corpus
File	autonomataTooDemo1.1.1.zip	autonomata_G2P_v61.tgz	N.v.t.	Corpus	dupira1.0.zip	Corpus	Corpus
Ontwikkelaar	UGent, RU Nijmegen, UUU, Nuance, TelAAtBas	UGent, RU Nijmegen, UUU, Nuance, TelAAtBas	SUG, CNTS - UA, ILK - UVT, Language and Computing	K.U.Leuven, RUG, O-gg	RU Nijmegen, Phasar-by	Tilburg Universiteit, RU Nijmegen, UA, Hogeschool Gent	INL, Van Dale, Selfservice Company, Knowledge Concepts
Omschrijving	Een demo van een service die automatisch spraak naar tekst herkent voor POIs (Points of Interest). Deze demo herkent overnachtingsadressen en eetgelegenheden in enkele grote steden (o.a. Amsterdam, Antwerpen, Gent, Rotterdam).	De AUTONOMATA-transcriptietoolset bestaat uit een transcriptie toolset voor de taal waarmee men woordenlijsten kan verkrijgen met nauwkeurige uitspraakinformatie. De tool maakt gebruik van een algemene grafische naar-foneemomzetter (de g2p-omzetter).	De COREA-coreferentieservice kan automatisch coreferente relaties tussen nominale constituenten in teksten oplossen.	De DAISY-demonstrator is een automatische samenvatter. Deze applicatie kan gebruikt worden voor test- en evaluatiedoeleinden. De samenvatter genereert drie soorten informatie: een samenvatting in de vorm van een krantenkop, een samenvattende tekst van de bron, en de definitie of procedure (of 'definite').	Dupira is een Dependency Parser voor het Nederlands, ontwikkeld aan de Radboud Universiteit in Nijmegen. Dupira is een regelgebaseerde parser, die uit de Dupira grammatica, lexicon en fact tables is gegenereerd met de AGFL parser generator. Het behelst van de Dupira parser de morfologische zinnen naar dependency graphs. Dupira is ontwikkeld voor praktische toepassingen in Information Retrieval en informatiesystemen die een natuurlijke taal-interface nodig hebben. De beoogde gebruikers zijn eerder computerwetenschappers dan taalkundigen. Meer documentatie is te vinden op http://www.cnts.ua.ac.be/fris/corea ("Abstrait-Based Dependency Parser for Dutch", Technisch Rapport. Dit document wordt samen met Dupira gedistribueerd.	Frog is een tokenizer, tagger, lemmatizer, morfologisch segmenter (gegmenteert een samengesteld woord in zijn onderdelen), shallow parser (herkent de niet-gebiede woorden in een zin), syntactische recognizer (herkent eigenaamen van o.a. personen, locaties en plaatsen), een dependency parser in één. Frog is een doorontwikkeling van Tadpole, voorheen Memory-Based Dutch Tagger-Lemmatizer. De tagger-lemmatizer is in eerste instantie ontwikkeld voor het project Corpus Gesproken Nederlands.	Interactieve online spellingshulp, waaraan men spellingsvragen in natuurlijke taal kan stellen: "hoe spel je pyama?". De spellingshulp is benaderbaar via de browser.
Soort licentie	Niet-commercieel = € 0	Niet-commercieel = € 500	Zie http://corea.tst-centrale.org/	Zie http://daisy.tst-centrale.org/	€ 0	Zie http://ilk.uvt.nl/frog/	Zie http://www.spelspiek.nl

10 Abbreviations and terms used

ABB	Company, dating back to 1883, concerned with power technology. http://new.abb.com/about/technology/corporate-research-centers/corporate-research-center-germany
ACM	Association for Computing Machinery http://www.acm.org
ACM SigSoft	ACM-Special Interest Group on Software Engineering http://www.sigsoft.org/
Android	Android is an open-source operating system for mobile phones, tablet computers, cameras etc., based on Linux and Java. https://www.android.com/
ANDS	Australian National Data Service http://www.ands.org.au/
Arcade	See Xbox
API	Application Programming Interface: a set of routine definitions, protocols, and tools for building software and applications.
ARDC	The Australian Research Data Commons (http://www.ands.org.au/guides/discovery-ardc). The term 'commons' traditionally refers to resources that are made available for community use. ANDS is creating a digital realisation of this concept for research data to enable greater data sharing and reuse.
B&G	Beeld en Geluid, the Netherlands Institute for Sound and Vision, is a cultural-historical organisation of national interest. It collects, preserves and opens the audiovisual heritage for as many users as possible: media professionals, education, science and the general public. www.beeldengeluid.nl/en
BnF	Bibliothèque Nationale de France (BnF) http://www.bnf.fr/en/tools/a.welcome_to_the_bnf.html
C	C, originally developed by Dennis Ritchie between 1969 and 1973 at Bell Labs, is a general-purpose, imperative computer programming language, supporting structured programming, lexical variable scope and recursion, while a static type system prevents many unintended operations. By design, C provides constructs that map efficiently to typical machine instructions, and therefore it has found lasting use in applications that had formerly been coded in assembly language, including operating systems, as well as various application software for computers ranging from supercomputers to embedded systems. See: https://en.wikipedia.org/wiki/C_(programming_language)
C++	C++ is a general purpose object oriented programming (OOP) language, developed by Bjarne Stroustrup and first published in 1985, and is an extension of C language. It is therefore possible to code C++ in a "C style" or "object-oriented style." In certain scenarios, it can be coded in either way and is thus an effective example of a hybrid language. It is considered to be an intermediate level language, as it encapsulates both high and low level language features. See: https://www.techopedia.com/definition/26184/c-programming-language (watch out for advertisement pop-ups)
CATCH	Continuous Access To Cultural Heritage (CATCH) and CATCHPlus are NWO-funded research programmes that ran from 2004-2012 (http://www.nwo.nl/en/research-and-results/programmes/Continuous+Access+To+Cultural+Heritage+(CATCH)) and later extended by http://www.catchplus.nl for another four years.
CCDD	Culturele Coalitie voor Digitale Duurzaamheid (Cultural Coalition for Digital Preservation) is a knowledge network of cultural heritage institutions aimed at collectively ensuring sustained access to digital information. http://www.nccd.nl/en/about-the-nccd/partners/cultural-coalition-digital-preservation/
CCS	Computer Conservation Society http://www.computerconservationsociety.org/ , is a cooperative venture between the British Computer Society, the Science Museum of London, and the Museum of Science and Industry in Manchester. The CCS was constituted in September 1989.
CD	Compact Disc (CD) is a digital optical disc data storage format, co-developed by Philips and Sony. The format was originally developed to store and play only sound recordings but was later adapted for storage of data (CD-ROM). Several other formats were further derived from these, including write-once audio and data storage (CD-R), rewritable media (CD-RW), Video Compact Disc (VCD), Super Video Compact Disc (SVCD), Photo CD, Picture-CD, CD-i, and Enhanced Music CD. Audio CD players have been commercially available since October 1982, when the first model was released in Japan. (From: https://en.wikipedia.org/wiki/Compact_disc)
CD-ROM	See: CD

CEA	Commissariat à l'énergie atomique et aux énergies alternatives (CEA) is a modern research institute organisation, consisting of various individual research facilities across France (http://www.cea.fr/english)
CeCILL	CEA-CNRS-INRIA-Logiciel Libre is a French based software license system (http://www.cecill.info/index.en.html).
CENDARI	Collaborative European Digital Archive Infrastructure (CENDARI) is a European FP7-supported project, led by Trinity College, Dublin, Ireland from 2012-2016 with the purpose to provide and facilitate access to existing archives and resources in Europe for the study of medieval and modern European history (specifically the First World War period) through the development of an 'enquiry environment' (http://www.cendari.eu/)
CKAN	Data Management System (Comprehensive Kerbal Archive Network (CKAN)) http://ckan.org/faq/
CeRCH	Centre for e-Research was established by King's in November 2007 in response to the Arts & Humanities Research Council's decision to withdraw funding from the AHDS at the end of March 2008. http://dh2010.cch.kcl.ac.uk/about/the-local-hosts-cch-and-cerch/centre-for-e-research-cerch.html
CIF21	Cyberinfrastructure Framework for 21st Century Science and Engineering (CIF21) is an NSF-programme (US). http://www.nsf.gov/funding/pgm_summ.jsp?pims_id=504730
CLARIAH-NL	Common Lab Research Infrastructure for the Arts and Humanities. CLARIAH, as a national counterpart to CLARIN and DARIAH, is directly and deeply embedded in the Europe-wide ESFRI enterprise. CLARIN and DARIAH are the only two humanities infrastructures under development on the ESFRI Roadmap, both projects were on the 2008 national roadmap, and they joined forces in CLARIAH, which was put on the 2012 national roadmap. The Netherlands has played an important role in establishing a consortium of national infrastructure projects united at the European level in the so-called CLARIN ERIC, which was established in February 2012 and is hosted by the Netherlands. www.clariah.nl
CLARIN (-D)	See for information the lemma over CLARIAH-NL. http://www.clarin-d.de/en/
CKAN	CKAN, The Comprehensive Kerbal Archive Network (CKAN), is an open-source data portal platform. It is a complete out-of-the-box software solution that makes data accessible – by providing tools to streamline publishing, sharing, finding and using data. www.Ckan.org
CMS	Content Management System. Usually a software environment that allows one to create and maintain websites.
CNRS	Centre National de Recherche Scientifique, the French national research council. http://www.cnrs.fr/index.php
CPU	Central Processing Unit, every computer has at least one.
CRAN	CRAN is a network of ftp and Web servers around the world that store identical, up-to-date, versions of code and documentation for R (see also: R) https://cran.r-project.org/
CRU	Computing Resource Unit (probably). Unit and system used by Renater (French Research and Education Network) for access to public (computing) resources. https://cru.renater.fr/sac/
CSC	Finnish national computing and data centre. http://www.csc.com/
CSESSP	Computational Science & Engineering Software Sustainability and Productivity Challenges (CSESSP Challenges). A programme by the US NITRD (The Networking and Information Technology Research and Development Program) https://www.nitrd.gov/csessp/
CSIRAC	First Australian (and fifth in the world) digital computer http://www.cs.mu.oz.au/dept/about/csirac/
Cuda	CUDA is a parallel computing platform and programming model invented by NVIDIA. It enables dramatic increases in computing performance by harnessing the power of the graphics processing unit (GPU). http://www.nvidia.com/object/cuda_home_new.html#sthash.Ay8XxBKq.dpuf
DANS	Data Archiving and Networked Services, a Dutch institute by NWO and KNAW, concerned with advanced data archiving (storing, enriching (metadata) and searching). Dans.knaw.nl
DARIAH (-EU)	DARIAH is a network connecting several hundreds of scholars and dozens of research facilities in currently 17 European countries, the DARIAH member countries. www.dariah.eu/
dBASE	dBase (also stylised dBASE), originally published by Ashton-Tate for microcomputer operating system CP/M in 1980, and later ported to Apple II and IBM PC computers

running DOS, was one of the first and successful database management systems for microcomputers. It includes a core database engine, a query system, a forms engine, and a programming language that ties all of these components together. dBase's underlying file format, the .dbf file, is widely used in applications needing a simple format to store structured data. <https://en.wikipedia.org/wiki/DBase> and www.dbase.com/

- DEC Digital Equipment Corporation, also known as DEC, and using the trademark, Digital, was a major American company in the computer industry from 1957 till 1998. Digital was a leading vendor of computer systems, including computers, software, and peripherals. Their PDP and successor VAX products were the most successful of all minicomputers in terms of sales. Digital was acquired in June 1998 by Compaq, which subsequently merged with Hewlett-Packard (HP) in May 2002. Some parts of Digital, notably the compiler business and the Hudson, Massachusetts facility, were sold to Intel. See <http://www.digital.com/> and https://en.wikipedia.org/wiki/Digital_Equipment_Corporation
- DEFF Danmarks Elektroniske Fag- og Forskningsbibliothek (DEFF) is an organisational and technological collaboration between academic, research and education libraries, which is co-funded by the Ministry of Higher Education and Science, the Ministry for Children, Education & Gender Equality, and Ministry of Culture in Denmark. www.deff.dk/english/
- DevLOG Réseau des acteurs du Développement LOGiciel au sein de l'Enseignement Supérieur et de la Recherche. A collaborative network of French researchers involved in the design, programming, distribution, maintenance of software within Higher Education and Research (ESR). <http://devlog.cnrs.fr/region>
- DFG Deutsche Forschungsgemeinschaft (The German Research Council). <http://www.dfg.de/en/>
- DOI International DOI Foundation (IDF), a not-for-profit [membership organisation](#) that is the governance and management body for the [federation of Registration Agencies](#) providing Digital Object Identifier (DOI) services and registration, and is the registration authority for the ISO standard (ISO 26324) for the DOI system. The DOI system provides a technical and social infrastructure for the registration and use of persistent interoperable identifiers, called DOIs, for use on digital networks. From: www.doi.org
- DOS Disc Operating System, practically synonymous with MS-DOS (for Microsoft's DOS). Originally developed in 1981 by Microsoft for IBM, and the standard operating system for IBM-compatible personal computers. The initial versions of DOS, a command line type OS, were simple but subsequent versions became increasingly sophisticated as they incorporated features of minicomputer operating systems. DOS is a 16-bit operating system and does not support multiple users or multitasking. Elements of DOS still survived today's MS Windows operating systems.
- DPC Digital Preservation Coalition (UK). <http://www.dpconline.org/>
- DRM Digital Rights Management (DRM) schemes are various access control technologies that are used to restrict usage of proprietary hardware and copyrighted works. DRM technologies try to control the use, modification, and distribution of copyrighted works (such as software and multimedia content), as well as systems within devices that enforce these policies. The use of digital rights management is not universally accepted. See https://en.wikipedia.org/wiki/Digital_rights_management
- Drupal A Open Source Content Management System (to create and maintain websites). Acronym derived from the words "dorp", "drop" and "druppel", Dutch words, as explained in the history section of the Drupal website. <https://www.drupal.org/about/history>
- DVD Digital Versatile Disc or Digital Video Disc is a digital optical disc storage format invented and co-developed by Philips, Sony, Toshiba, and Panasonic in 1995. The medium can store any kind of digital data and is widely used for software and other computer files as well as video programs watched using DVD players. DVDs offer higher storage capacity than compact discs while having the same dimensions.
- DVD-ROM A new type of read-only compact disc that can hold a minimum of 4.7GB (gigabytes), enough for a full-length movie. The DVD-ROM specification supports disks with capacities of from 4.7GB to 17GB and access rates of 600 KBps to 1.3 MBps. One of the best features of DVD-ROM drives is that they are backward-compatible with CD-ROMs. This means that DVD-ROM players can play old CD-ROMs, CD-I disks, and video CDs, as well as new DVD-ROMs. Newer DVD players can also read CD-R disks. DVD-

	ROMs use MPEG-2 to compress video data. http://www.webopedia.com/TERM/D/DVD_ROM.html
EC	European Commission. http://ec.europa.eu/index_en.htm
ECP	Electronic Commerce Platform Nederland (ECP) is a Dutch Platform for the information society, in which government, business and social organisations cooperate and exchange knowledge for the deployment of information and communication technology in Dutch Society. https://ecp.nl/ (Dutch only)
EIF	The Education Investment Fund (EIF) was established by the <i>Nation-building Funds Act (Australia) 2008</i> . It replaced the Higher Education Endowment Fund (HEEF), which was established in 2007. The EIF was initially credited with \$6.48 billion. https://www.education.gov.au/education-investment-fund
EMC	EMC is an US-based company, focusing on large scale storage and cloud solutions and founded in 1979 by Richard Egan and Roger Marino (the E and M in EMC). http://www.emc.com/en-us/index.htm
ePLAN	Platform of eScience/Data Research Centers in Europe, an as yet informal network in the Netherlands. escience-platform.nl
eSTeP	eScience Technology Platform, an organisational module by NLeSC, to store, distribute and exchange eScience-qualified software. estep.esciencecenter.nl
EWI	"Elektrotechniek, Wiskunde en Informatica": the Faculty for electrical engineering, mathematics and computer science of Delft University of Technology. http://www.ewi.tudelft.nl/en/
EZ	"Economische zaken": the Dutch Ministry of Economic Affairs. www.minez.nl
Flash	Flash is a multi-media tool and programming environment developed and distributed by Adobe Systems Incorporated. Flash (sometimes: Macromedia Flash or Shockwave Flash) is used for the production of animations, browser games, rich Internet applications, desktop applications, mobile applications and mobile games. Flash displays text, vector graphics and raster graphics to provide animations, video games and applications. It allows streaming of audio and video, and can capture mouse, keyboard, microphone and camera input. https://en.wikipedia.org/wiki/Adobe_Flash (http://www.adobe.com/software/flash/about/)
FLOSS	FLOSS stands for "Free/Libre and Open Source Software". Compare to "FOSS," which stands for "Free and Open Source Software." The term Libre (from French) emphasises the freedom in use, not freedom in cost ("for free"). In that sense the term FLOSS is more concise. https://www.gnu.org/philosophy/floss-and-foss.en.html
FP7	Seventh Framework Programme for Research and Technological Development (2007-2013). The EU works with multi-annual work programs to boost science and economy. FP7 refers to the seventh of such programs, with its official legislation, application and report structures and funded projects. http://cordis.europa.eu/fp7/
FTE	Full Time Equivalent. Unit of personnel employment (one person-year)
FSF	The Free Software Foundation (FSF) is a non-profit organisation with a worldwide mission to promote computer user freedom. Founded in 1985, FSF defends and promotes computer users' right to use, study, copy, modify, and redistribute computer programs. The FSF promotes the development and use of free (as in freedom) software. http://www.fsf.org/press
Git	Git is a version control system created by Linus Torvalds (the guy that created Linux), used for software development and other version control tasks. As a distributed revision control system it is aimed at speed, data integrity, and support for distributed, non-linear workflows. See https://en.wikipedia.org/wiki/Git_(software)
GitBlit	Gitblit is an open-source, pure Java stack for managing, viewing, and serving Git repositories. It's designed primarily as a tool for small workgroups who want to host centralised repositories. Gitblit.com
Github	GitHub is a Web-based Git repository hosting service. It offers all of the distributed revision control and source code management (SCM) functionality of Git as well as adding its own features. Unlike Git, which is strictly a command-line tool, GitHub provides a Web-based graphical interface and desktop as well as mobile integration. It also provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project. GitHub offers both plans for private repositories and free accounts, which are usually used to host open-source software projects. See: https://en.wikipedia.org/wiki/GitHub and https://github.com
Glm	Generalised linear model (GLM) is a flexible generalisation of ordinary linear regression. Some consider it a statistical model, others consider it as a machine learning model. Both these elements are not mutually exclusive (see:

	http://datascience.stackexchange.com/questions/488/is-glm-a-statistical-or-machine-learning-model)
GNU	GNU (GNU's Not Unix) is a free and open source operating system with lots of adjacent software. https://www.gnu.org/home.en.html
GPL(-3)	General Public License (GPL) is a Copyleft license. Copyleft is a framework that permits ongoing sharing of a published work, with clear permissions that <i>both</i> grant <i>and</i> defend its users' freedoms — in contrast to other free licenses that grant freedom but don't defend it. https://sfconservancy.org/copyleft-compliance/principles.html , https://www.gnu.org/licenses/gpl-3.0.en.html
HSN	The Historical Sample of the Netherlands (HSN). The HSN offers a representative sample of about 78.000 people born in the Netherlands during the period 1812-1922. The HSN-database containing individual life-courses is a unique tool for research in Dutch history and demography. https://socialhistory.org/en/hsn/index
HTML (-n)	Hyper Text Mark-up Language is the elementary man-readable language, used to format and visually represent the contents of a webpage. HTML adds "mark-up" to basic text not only by adding graphics, tables and (bold, italic, fonts, -sizes, etc.) but also (hyper-)links to other Web pages, or documents. The -n reflects that there are subsequent versions of the language since its introduction. https://www.w3.org/html/ ; https://developer.mozilla.org/en-US/docs/Web/HTML
Http(s)	Hyper Text Transfer Protocol (HTTP) is an application protocol that basically executes the link references in HTML. Whenever a link is clicked or a website's address typed in manually in a browsers address field, HTTP executes the command and diverts the browser to show the addressed page. Development of HTTP was initiated by Tim Berners-Lee at CERN in 1989. Since its introduction a few newer versions have been defined, among which "Https", a secure version that encrypts the data passed. https://www.w3.org/Protocols/
I/O	Input/Output
IBM	Originally: International Business Machines (Cooperation), a multinational company involved in basically all aspects of designing, manufacturing and selling calculating machines and service provision. The name originates from 1924, the underlying merged companies date back to 1911. www.ibm.com
ICJ	The Camille Jordan Institute is a French Research Department in Mathematics located at several institutes at multiple locations. http://math.univ-lyon1.fr/?lang=en
IdGC	L'Institut des Grilles et du Cloud (IDGC) is a CNRS infrastructure organisation in charge of the coordination and execution of CNRS research activities in the domain of grid and cloud production. http://idgc.in2p3.fr/en/
IDS	The Intermediate Data Structure (IDS) was developed to try to solve longitudinal historical micro-level demographic data challenges for research by facilitating the storing and sharing of such data. http://www.ehps-net.eu/content/ids
IEEE	The Institute of Electrical and Electronics Engineers, dates back to 1963 is a US-based professional organisation active in the domain of Electrical, Electronics, Communications, Computer Engineering, Computer Science and Information Technology with a global scope and over 400000 members worldwide. See www.ieee.org . Its objectives are (according to its website) to foster technological innovation and excellence for the benefit of humanity.
IISG or IISH	Dutch International Institute for Social History (<i>Internationaal Instituut voor Sociale Geschiedenis</i>). Conducts research and collects data on the global history of labour relations. https://socialhistory.org/
INL	Institute for Netherlands Lexicology. http://www.inl.nl/our-work-and-working-methods
INRA	Institut national de la recherche agronomique, French National Institute for Agricultural Research. An agricultural research institute and centre for the agricultural sciences, focusing on solutions for society's major challenges. http://institut.inra.fr/en
Inria	Institut national de recherche en informatique et en automatique is a French national research institution focusing on computer science and applied mathematics. It is a public science and technology institution dedicated to computational sciences and promotes "scientific excellence for technology transfer and society". http://www.inria.fr/en/
Inseit	<i>Instituut voor Supervisie op Enterprise IT & Innovatie</i> is a Dutch organisation that aims to strengthen corporate governance on digital innovation. inseit.nl (Dutch only)
Inserm	Institut national de la santé et de la recherche médicale. Founded in 1964, the French National Institute of Health and Medical Research (Inserm) is a public scientific and technological institute which operates under the joint authority of the French Ministry

	of Health and French Ministry of Research. http://english.inserm.fr/about-inserm/the-institute-missions
iOS	iOS “iPhone Operating System” is Apple’s operating system for Apple (mostly mobile) hardware: iPhone, iPad, and iPod touch and iPad Mini. Today’s actual version is 9, while iOS10 has been announced for fall 2016. http://www.apple.com/ios/ios10-preview/
IPR	Intellectual Property Rights concern the protections granted to the creators of intellectual property (artistic works, text phrases, symbols, texts, software, etc.) and may involve trademarks, copyrights, patents and industrial design rights.
ISO	International Organisation for Standardisation is an independent, Switzerland-based non-governmental international organisation with over 160 national standards bodies. Through its members, it brings together experts to share knowledge and develop voluntary, consensus-based, market relevant International Standards that support innovation and provide solutions to global challenges. ISO issues certificates to organisations that have proven to follow one of their standards. www.iso.org
IT	Information Technology (in Europe, also: ICT, or Information and Communication Technology).
ITIL	ITIL is a best practice framework drawn from both the public and private sectors internationally. It describes how IT resources should be organised to deliver business value, documenting the processes, functions and roles of IT Service Management (ITSM). https://www.axelos.com/best-practice-solutions/itil
Java	Java is a general-purpose computer programming language -available as a free download- developed by SUN Microsystems and introduced in 1995. It is designed to have as few implementation dependencies as possible so that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. https://www.oracle.com/java/index.html and https://en.wikipedia.org/wiki/Java_(programming_language)
Java JRE	Java Runtime Environment is an implementation of the Java Virtual Machine which actually executes Java programs. It is a plug-in needed for running java programs. https://www.java.com/en/download/faq/techinfo.xml
Javascript	Is a scripting language, largely complementary to Java, with which it only shares part of the name. It was made public in 1995 by Netscape who had ordered its development. It is now widely used. https://en.wikipedia.org/wiki/JavaScript
JDEV	Productive Java-based Application Development (Oracle JDeveloper) and a free integrated development environment that simplifies the development of Java-based applications. http://www.oracle.com/technetwork/developer-tools/jdev/overview/index-094652.html
Jenkins	Jenkins is an open source continuous integration tool. This means it is a tool to support merging all developer working copies of a software development to a shared mainline frequently (up to several times a day). It is written in Java. https://jenkins.io/
Jisc	Originally: Joint Information Systems Committee (UK), that supports higher education and research by providing advice, digital resources and network and technology services. (For the Netherlands: compare to SURF). Jisc is funded by a combination of the UK further and higher education funding bodies, and individual higher education institutions. https://www.jisc.ac.uk/
KB	Koninklijke Bibliotheek (National Library of the Netherlands). https://www.kb.nl/en
KE	The Knowledge Exchange (KE) partners are five key national organisations within Europe tasked with developing infrastructure and services to enable the use of digital technologies to improve higher education and research: CSC in Finland, DEFF in Denmark, DFG in Germany, Jisc in the UK and SURF in the Netherlands. www.knowledge-exchange.info
KNAW	Koninklijke Nederlandse Academie van Wetenschappen. The Royal Dutch academy of Sciences. https://www.knaw.nl/en
LAAS	Laboratory for Analysis and Architecture of Systems, a CNRS Research Institute (Fr). https://www.laas.fr/public/en
LIGM	Laboratoire d'Informatique Gaspard-Monge, a French informatics (Computer Science) institute belonging to PRES Université Paris-Est (UPE), located at the campus of Champs-sur-Marne . http://ligm.u-pem.fr/
LIMA	Founded by experts from the former NIMk (Nederlands Instituut voor Mediakunst (http://www.nimk.nl/)), LIMA is the international platform for sustainable access to media art. With knowledge and passion for both art and technology, these experts are

	insuring that video, digital and performance artworks can and will be presented now and in the future. http://www.li-ma.nl/site/
Linux	Linux is a Unix-like, open-source operating system (OS) of which the kernel was written and published in 1991 by Linus Torvalds. Linux.org
LP	Long Playing record: an analogue sound storage medium (on vinyl), in use since 1948, which is still being used for storing and reproducing music. http://lprecord.umwblogs.org/history/invention/
Mac OS-X	OS X is a series of Unix-based graphical interface operating systems (OS) developed and marketed by Apple Inc. It is designed to run on Macintosh computers and forms the second most used OS worldwide. It is the 1999 follow-up of the original series of OS (up to version 9) that started in 1984. https://en.wikipedia.org/wiki/OS_X
MS-DOS	Microsoft Disc Operating System) is a discontinued operating system for x86-based personal computers mostly developed by Microsoft, originally for IBM (1981) and was the main operating system for IBM PC compatible personal computers during the 1980s and the early 1990s and now superseded by the graphical Microsoft Windows operating system. https://en.wikipedia.org/wiki/MS-DOS
NCDD	Nederlandse Coalitie voor Digitale Duurzaamheid (Netherlands Coalition for Digital Preservation) is a foundation funded and supported by the Dutch ministry of Science, Culture and Education and with the following members: DANS, KB, NA, B&G and the CCDD. http://www.ncdd.nl/en
NDE	Netwerk Digitaal Erfgoed, Network Digital Heritage. Cooperation between KB, B&G, RCE, KNAW and NA, partners concerned with digital heritage. http://www.den.nl/pagina/511/netwerk-digitaal-erfgoed/ (in Dutch)
NDL	Naive discriminative learning. Name of a software package that enables the user to model a learning process based on single events using the learning model of Rescorla-Wagner or predicting the final state of learning using the equilibrium equations by Danks. NDL calculates weights and activations between a set of discriminative features (cues) and their outcomes which they activate. http://frequenz.uni-freiburg.de/770
NDL2	Newer version of NDL.
Nuxeo	
Nexus	Nexus is a deposit management platform, allowing to host artefacts. These artefacts are components generated e.g. to build a project, and then deposited on Nexus. Nexus' interest is to be able to share artefacts with other project developers up to an entire community. Look for "SourceSup nexus" to find https://services.renater.fr/sourcesup/nexus in order not to be confused with any Google-related products with the name Nexus.
NLeSC	Netherlands eScience Center, a Dutch public body (foundation) funded and founded by NWO and SURF. esciencecenter.nl
NREN	National Research & Education Network. Name in use for national data communication networks for science and education (usually publicly funded). https://en.wikipedia.org/wiki/National_research_and_education_network
NSF	National Science Foundation, the US national funding organisation. https://www.nsf.gov
NTU	Nederlandse Taalunie or Dutch Language Union. Based on a 1980 treaty between the Netherlands and Belgium (Flanders), with Surinam as an associated member and collaborations with Aruba, Curaçao and Sint Maarten to define (and keep up to date) and preserve the Dutch language.
NWO	Netherlands Organisation for Scientific Research, the Dutch national funding organisation. www.nwo.nl
OCW	Ministerie van Onderwijs, Cultuur en Wetenschap: the Dutch Ministry of Science, Culture and Education. www.minocw.nl
OKF	Open knowledge Foundation (http://okfn.org)
OMII-UK	Open Middleware Infrastructure Institute. OMII-UK enables advanced e-infrastructure solutions based on open source Grid middleware components which are engineered to a high quality, interoperable and easily-used, drawing on the software generated by the UK e-Science programme. OMII-UK is an expansion of the Open Middleware Infrastructure Institute (OMII) that was originally established as a 3 year project in Southampton in January 2004. http://gow.epsrc.ac.uk/NGBOViewGrant.aspx?GrantRef=EP/D076617/1
OS	Operating System
OSI	Open Source Initiative. Development based on the sharing and collaborative improvement of software source code. OSI was formed in 1998 as an educational,

advocacy, and stewardship organisation at this important moment in the history of collaborative development. <https://opensource.org/>

P.I.	Principal Investigator
PC	Personal Computer, usually also a Portable Computer. A small size computer for use by an individual without external other computers.
PDP	Programmed Data Processor (PDP) was a series of minicomputers designed and manufactured by the Digital Equipment Corporation from 1957 to 1990. Many of series (PDP-1 up to PDP-16) were commercially successful and some rather innovative products.
PID	Persistent Identifier. A code (or address), usually directly recognised by Web browsers in the address field, issued by several organisations, to identify and locate digital objects over the Internet. Whereas Internet links may disappear over time, Persistent Identifiers ensure the long term addressability and accessibility of digital objects. http://www.pidconsortium.eu/
PhD	Doctor of Philosophy. The doctorate is minimally based on 3 or more years of graduate study and a defended dissertation; the highest degree awarded graduate study. Equivalent to the title "dr."
PL/I	Programming Language One (Roman I) is an imperative computer programming language designed by and for IBM's IBM System/360 in the sixties, a computer meant for both business and scientific calculations. The language is still in use. http://www-03.ibm.com/software/products/en/plicompfam
PLAN-E	PLATform of National eScience Centers in Europe. A cooperation by major eScience (data science, computational science) centres in over 20 European countries. plan-europe.eu
Plume	Preuves et Langages, Un Manège Enchanté is a computer science research ensemble. The research carried out by the members of the Plume team is mostly centered on two strongly intertwined themes: <i>logical foundations of programming languages</i> and <i>theory of computing systems</i> . http://www.ens-lyon.fr/LIP/PLUME/
RABO	RABObank, is a Dutch cooperative bank originating from credit banks for farmers, from locally raised savings. In this document the link is with the RABO Art Collection. https://www.rabobank.com/nl/about-rabobank/profile/history/index.html
RAD	Rapid Application Development is a software development methodology based on a development process rather than on strong a priori specifications. Experience gained throughout the creation of the product is used to improve the product. Particularly suited for user interfaces. http://www.tutorialspoint.com/sdlc/sdlc_rad_model.htm
RBDD	Réseau Bases de Données, Database Network. A CNRS community Based on two surveys on data bases in use and the issues related to them (including sustainability). rbdd.cnrs.fr
RCCP	R and C++ ("plus plus") integration package. Dirk Eddebuettel and Romain Francois (2011). Rcpp: Seamless R and C++ Integration. Journal of Statistical Software, 40(8), 1-18. URL http://www.jstatsoft.org/v40/i08/ and Eddebuettel, Dirk (2013) Seamless R and C++ Integration with Rcpp. Springer, New York. ISBN 978-1-4614-6867-7.
RCE	Rijksdienst voor het Cultureel Erfgoed, Ministerial service organisation for cultural heritage, a department of the Ministry of Science, Culture and Education. http://cultureelerfgoed.nl/
RENATER	Réseau national de télécommunications pour la technologie, l'enseignement et la recherche (National telecommunications network for Technology, Education and Research) is the national research and education network (NREN) in France, founded in the beginning of the nineties. https://www.renater.fr/?lang=en
REST-API	REpresentational State Transfer (REST)- Application Programming Interface. REST is a software architectural style used in World Wide Web applications and consists of a set of style elements that implicitly impose some restrictions to building webservices/interfaces (compared to all is free). An API build by using REST as basic framework is referred to as REST(ful) API.
SCEN	Stichting Computer Erfgoed Nederland (SCEN) (Foundation for Computer heritage). Website presently being renewed. (http://www.computer-erfgoed.nl/)
SHFT(-nn)	Series of born digital art programs written by Peter Struycken. http://www.li-ma.nl/site/article/shft-34-1982-2007-peter-struycken
SI ²	Software Infrastructure for Sustained Innovation - SSE & SSI (SI ² : SSE & SSI), an NSF programme for funding dedicated to sustainable software (developments) https://nsf.gov/funding/pgm_summ.jsp?pims_id=503489
Sympa	Système de multipostage automatique, a Renater hosted mailserver. https://www.renater.fr/logiciel-sympa

Sonar	Sonar is an open source Platform used by development teams to manage source code quality. Sonar has been developed to make code quality management accessible to everyone with minimal effort. Since version 2.0, Sonar enables to cover quality on 7 axes and so to report on Duplicated code, Coding standards, Unit tests, Complex code, Potential bugs, Comments, Design and architecture. http://www.sonarqube.org/
SourceSup	SourceSup is a project management Web platform for Higher Education and French Research organisations. Every member of the community can create a project on SourceSup. SourceSup is hosted by Renater. https://sourcesup.cru.fr/
SME	Small and Medium Enterprise, defined in the EU recommendation 2003/361 on the basis of staff headcount and either turnover or balance sheet total. http://ec.europa.eu/growth/smes/business-friendly-environment/sme-definition_en
SQL	Structured Query Language is a programming language for managing and processing data held in a relational database management system. It is an ANSI and ISO standard since the mid-eighties. http://www.w3schools.com/sql/sql_intro.asp
SSI	Software Sustainability Institute, an Academic institute in the UK, providing knowledge and support regarding software sustainability matters. It is supported by several universities in the UK and linked to Edinburgh's Parallel Computing Centre. SSI is also being used recently for Software Sustainability Initiative and Software Sustainability Infrastructure. https://www.software.ac.uk/
SSoA	Software Seal of Approval, a development similar to the Data Seal of Approval, to encourage and reward trends towards writing software properly from the point of view of software sustainability. www.softwaresealofapproval.org .
STATA	STATistical data Analysis, name of a Company (www.stata.com) and a software package for statistical data analysis.
Subversion	Apache Subversion (SVN) is a software versioning and revision control system distributed as free software, to maintain current previous versions of documents (mainly source code, Web pages), started in 2000. https://subversion.apache.org/
SUN	Sun Microsystems, Inc. was a versatile computer company that created in addition to hardware systems, lots of software still in use today (Such as Java, NFS (Network File System)). It was founded in 1982 (with SUN being a reference to the Stanford University Network (following wikipedia)) and acquired by Oracle in 2010, Sun was acquired by Oracle Corporation. https://www.oracle.com/sun/index.html
STEVIN	Subsidy Programme 2004-2010 for research and development in language and speech technology. http://www.kennislink.nl/publicaties/stevin-2004-2010
SURF	Originally "Samenwerkende Universitaire RekenFaciliteiten", today: Collaborative organisation for ICT in Dutch education and research. An organisation for and by Dutch universities, vocational high schools and research organisations, in the domain of ICT and infrastructure. Its scope compares to Jisc in the UK. Among other things responsible for the national computing and networking infrastructure for research and education. Two of its sub subsidiaries are SURFnet and SURFsara. www.surf.nl
TiMBL	Tilburg Memory-Based Learner Packages, a machine learning system developed at Tilburg University (NL). It is a fast, tree-based implementation of k-nearest neighbour classification. https://languagemachines.github.io/timbl/
TLTn	Treebanks and Linguistic Theories, name of an International Workshop on this topic. The number (n) refers to the number of the (annual) workshop. From: http://tlt14.ipipan.waw.pl/ ne can find back all previous TLT-events, back to nr. 1 in 2002.
TNO	<i>Nederlandse organisatie voor Toegepast Natuurwetenschappelijk Onderzoek</i> (Netherlands Organisation for Applied Physics research), which was founded in 1932 to enable business and government to apply knowledge. As an organisation regulated by public law, TNO is independent: not part of any government, university or company. www.tno.nl/en/
Triple	IN the context of this document, a triple is a data entity composed of subject-predicate-object, like "Bob is 35" or "Bob knows Fred".
Triplestore	A triplestore or RDF store is a purpose-built database for the storage and retrieval of triples.
TST	<i>Taal- en Spraaktechnologie (Centrale)</i> . This entity is ceasing to exist. Its contents will be temporarily stored at INL, the Institute for Netherlands Lexicology, which is set to merge into the Institute for the Dutch Language (<i>Instituut voor de Nederlandse Taal</i>). Original website: http://tst-centrale.org , now at http://www.inl.nl/
TSTC	TST Centrale. See TST.
TV	Television, a telecommunication medium and device, introduced to the public in the fifties, now a mass medium.

Ubuntu	Name of a Southern African philosophy, used for an open source software Linux platform dating back from 2004, that runs from the cloud, to the smartphone, including personal computers, and network servers. www.ubuntu.com
UDF	Universal Disc Format is an open file system for data storage mostly used for DVDs optical disc formats, UDF is developed and maintained by the Optical Storage Technology Association (OSTA). http://searchstorage.techtarget.com/definition/Universal-Disk-Format
UI	User Interface
Unity3d	Unity3d is a development platform for creating games and interactive 3D and 2D experiences like training simulations, medical and architectural visualisations, across mobile, desktop, Web, console and other platforms. Unity3d is a commercial globally operating company. www.unity3d.com
UNESCO PERSIST	A UNESCO activity which, among other matters will try to engage globally operating software companies (such as Microsoft, IBM, Google, Apple, etc.) to allow their obsolete software, including operating systems, to be used for heritage related purposes at fair conditions. http://www.unesco.org/new/en/media-services/single-view/news/persist_unesco_digital_strategy_for_information_sustainability/back/9597/#.V6xJGzWIVZE
URL	Universal Resource Locator is a standard formatted scheme to locate an object on the Internet. If the protocol referenced in the URL happens to be "http", the URL is identical to the not defined term "Web address". http://www.webopedia.com/TERM/U/URL.html
US	United States (of America)
USB	Universal Serial Bus. Since the mid-nineties a standardised connector type with the software to make the connect link work. A USB-stick is synonymous to a data storage medium with a USB connector. https://en.wikipedia.org/wiki/USB
UVA	University of Amsterdam. One of the two main universities in Amsterdam. www.uva.nl
UVC	A Universal Virtual Computer (UVC) is a virtual machine (VM) specially designed for preservation of digital objects such as held by libraries, archives and institutions alike. The method is based on emulation but does not require hardware or full emulation. https://en.wikipedia.org/wiki/UVC-based_preservation
Virtuoso	Virtuoso is a modern enterprise grade solution for data access, integration, and relational database management (SQL Tables and/or RDF based Property/Predicate Graphs) provided by the company Openlinksoftware. http://virtuoso.openlinksw.com/
VLC	VideoLan Client, an open source portable software product to play video files on computers. http://www.videolan.org/vlc/index.html
WiiWare	WiiWare is a service provided by Nintendo (computer games company) that allows Wii users to download games and applications for the Wii video game. https://www.nintendo.co.uk/Wii/WiiWare/WiiWare-Download-Games-on-Wii-Wii-Nintendo-UK-621696.html
WNT	<i>Woordenboek der Nederlandsche Taal</i> (Dictionary of the Dutch Language)
xBASE	Xbase is a loose reference to dBASE-like database languages and products.
Xbox	Xbox Live Arcade (XBLA) is Microsoft's Xbox Games Store, a download site for games from various publishers and game developers www.xbox.com/livearcade
XML	eXtensible Mark-up Language is a mark-up language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable, defined by the W3C's XML 1.0 Specification (and subsequent additions/updates). http://www.w3schools.com/xml/
XS4ALL	"Access for all", a major Dutch Internet Service Provider with a long history (from 1993) and originating from the anarchistic state the Internet developed from in the early nineties. https://www.xs4all.nl/over-xs4all/wie-wij-zijn.htm
Zenodo	Zenodo builds and operates a simple and innovative open digital repository that enables researchers, scientists, EU projects and institutions to share, preserve and showcase multidisciplinary research results (data and publications) that are not part of the existing institutional or subject-based repositories of the research communities. http://zenodo.org/

11 Article from *De Telegraaf* (in Dutch)

DEN HAAG - Gemeenten hebben problemen met het archiveren van informatie die op verouderde wijze is opgeslagen. Het gaat onder meer om materiaal op floppy's, cd-roms en taperecorders. Deze informatie blijkt inmiddels moeilijk vindbaar, niet doorzoekbaar en is soms in zijn geheel niet toegankelijk. Dat blijkt uit een steekproef van het vakblad Binnenlands Bestuur onder 26 gemeenten.

Aan het onderzoek deden gemeenten als Rotterdam, Utrecht, Breda en Eindhoven mee. Dertien eindverantwoordelijken voor het gemeentelijk archief geven aan dat ze soms of vaak moeilijkheden ondervinden bij het zoeken naar informatie. Daarnaast zeggen vijftien gemeenten informatie te missen. Twee van de ondervraagde gemeenten kampen bovendien met bitrot in hun archiefmateriaal. Door dit verschijnsel dooft de elektrische lading van een bit langzaam waardoor programma's niet meer werken. Vooral data op tapes blijken hier gevoelig voor.

Ongeveer de helft van de geïnterviewden vindt dat gemeentemedewerkers zich niet voldoende bewust zijn van het belang van goede archivering. Zij beoordelen hun kennis hierover als matig of onvoldoende. Het Nationaal Archief zegt tegenover Binnenlands Bestuur bekend te zijn met „de uitdagingen van digitale duurzaamheid“. De organisatie wil daarom kennisontwikkeling over dit onderwerp stimuleren.

Informatie die gemeenten willen bewaren, moet volgens de Archiefwet na maximaal twintig jaar verhuisd zijn naar bepaalde digitale bewaarplaatsen. In het verleden is ervoor gekozen materiaal op te slaan op inmiddels verouderde software en hardware. Nu blijkt die informatie grotendeels onleesbaar te zijn.