Referencing of complex software environments as representations of research data

Sven BINGERT [a,1], Stefan BUDDENBOHM [b] and Daniel KURZAWE [c]
[a] *Gesellschaft für wissenschaftliche Datenverarbeitung Göttingen*
[b] *Max-Planck-Institut zur Erforschung multiethnischer und multireligioser Gesellschaften*
[c] *Niedersächsische Staats- und Universitätsbibliothek Göttingen*

Abstract. Complex software environments, like virtual research environments or visualisation frameworks, are increasingly used to conduct research and present its results. While there is a growing amount for solutions facilitating the (granular) ci- tation of publications and research data, the citation of complex software environments remains a challenge. This abstract outlines the challenges and introduces an approach for referencing software environments developed in the Humanities Data Centre project: the application preservation.

Keywords. digital humanities, complex software environment, application preservation

1. Introduction

Progress and transparency in science largely depends on the capability of researchers to cite and reference the various aggregations of research data, instruments, and publications. Regardless what the subject of referencing may be, it is inevitable to sustain access to stable object representations that have to be documented in a transparent and proper way. For this end standards and infrastructure are necessary to serve the discipline- specific procedures of citing and referencing. Whereas these standards and infrastructure are quite established and harmonised for publications of research results - a traditional duty of libraries - the field is only developing with regard to research data. Beyond this an comprehensive overview and discussion of the evolving landscape of research data types and some implications can be found in [Sahle & Kronenwett 2013]. The challenges re- lated to referencing these new types of data will be described in detail in this paper, fo- cussing on complex software environments as representations of research data. The pro- posed solution revolves around an adapted persistent identification (PID) [Kalman 2015] approach applying fragment identifier and template handles. The insights base to a large part on the design phase of the Humanities Data Centre2 , a research data centre for the Humanities currently under construction.

1 Corresponding author. Gesellschaft fuer wissenschaftliche Datenverarbeitung Goettingen

2 http://www.humanities-data-centre.org; last visited March 2016

2. What do researchers reference?

The short answer is: Everything. There is no hard limitation regarding the objects of research. Every distinguishable object might be addressed and referenced or cited [Kalman 2015]. The range of objects is almost unlimited: files in a file system, database entries, web sites, books, places, people or journal articles. For many of those object classes solutions in form of services or tools to create and resolve references are available. The most prominent examples are ISBN3 for books or DOIs [Paskin 2010] for digital publications, but also unique stable references to people realised by ORCID4 are prevailing. However the unsolved problem remains, that a large share of research data does not fit in these categories. Despite this the heterogeneity with regard to size, format or structure doesn't make it easier to handle. So far for static objects as representations of research data. But what lies beyond these conventional, static object classes? It is obvious that researchers want to reference a broad range of research data types that do not fit into the static definition and that are only evolving. The development of new content or data types is closely aligned to the development of the working environment, methods and instruments of the researchers and for this reason quite difficult to be foreseen for an infrastructure provider. Also the citation of research data fulfils various purposes ranging from impact and reputation to transparency and reproducibility of research results so the scope of the archives can be widened. The latter point for example makes the archiving of different aggregations of one same data set interesting as it allows reproducing the certain processes. So in terms of content types not only conventional formats of data should be taken into account but also software environments, virtual research environments5 , complex databases6 , visualisation frameworks7 , collections, or processed data. As a common thread appears the complex character of these kinds of data, meaning that they can have various consecutive layers, aggregations or components. To some extent software is depending on its environment, but also on its way of usage. Both is evolving over time, also ontologies, terms or references in data bases. The state of software environments, like an interactive visualisation tools is also highly fluid but is, as we argue, also a research object by itself. In which way can these classes of objects be referenced? A reference will likely point to a specific fixed state of the object such as a query term or search string and not to the virtual research environment or the database as such. A workaround could be a reference pointing to a jump page of a research data centre or a repository providing the query term or search string. This can work out as long as the database remains stable. Out of the question is the inconvenience of this procedure as it requires additional steps in the absorption process on side of the reader. Therefore a more convenient solution is needed. This solution must provide a citation which includes descriptors of such a quality that the reference can point to clearly one specific data set, ideally integrated in one PID that just has to be resolved by the reader.

3 DIN ISO 2108
4 http://www.orcid.org; last visited March 2015
5 e.g. http://textgrid.de; last visited March 2016
6 e.g. http://www.berliner-klassik.de; last visited March 2016
7 .g. http://media.mpi.mmg.de, last visited March 2016

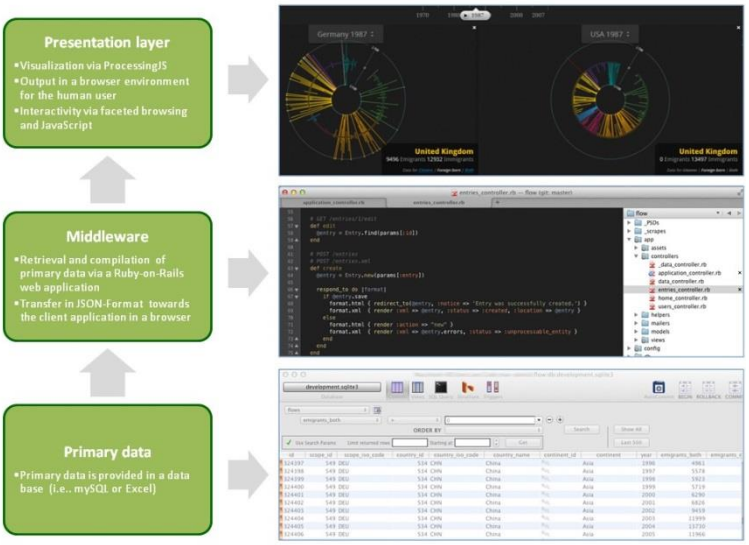

Figure 1. The multi layer character of a complex software environment using the example of the visualisation of global migration flows.

2.1. Excursus: visualisation framework

As an illustration for the above mentioned new object classes we describe a visualisation framework because it not only demonstrates the fluid character of the data but also what additional layers and dependencies have be taken into account when archiving and referencing this kind of data. For example there can be questions of a technical nature (granularity of the reference) or legal questions (licence status of content components) that influence the proposed infrastructural solution. The term visualisation framework may be seen in this specific context as synonym for a complex software environment. Other embodiments of complex software environments may be digital editions or virtual research environments. Our visualisation framework in this context [Aschenbrenner et.al. 2015] can be characterised as an attractive presentation of research data with interactive components aimed at the human user. Basically it is a database-service visualising the result of a search string. The illustration in Figure 1 depicts the Global Migration Flows8 , allowing the user to create individual data sets visualising migration movements between selected countries and over selected periods of time. The visualisation framework is accessible via a common browser and is based on data from the United Nations Population Division ranging from 1970 to 2011. The data visualisation is closely bound to its presentation environment, therefore an archival solution for this kind of data has to address this multiple layer character to sustain its added value. Usually this kind of data visualisation, based on a browser as access interface, consists at least of three layers (c.f. Figure 1):

8 http://flow.mmg.mpg.de; last visited March 2016

1. The normalised and enriched data provided through a database (data layer or primary data), which can be as simple as an Excel chart or ranging to more complex forms of databases.

2. A processing layer **(middleware)** that transfers the normalised and enriched data to the client application of the end user, the web browser. The necessary resources for the development and coding of the processing layer and the following presentation layer cannot be provided in each research project because of spare competences or resources. This advocates the refuse of these kind of application and data for other research projects.

3. On this rests the user interface, usually a common web browser. This presentation layer is normally out of the scope of action of the researcher. The visualisation framework has no influence on the browsers used by the end used and only can try to cover a range of most common standards.

The above standing remarks allow us to identify two overarching aspects that have to be addressed by any technological solution: the sustainability of the system (c.f. Sec. 3) and the citation of actual states of the system (c.f. Sec. 4).

3. Challenges related to the sustainability of complex software environments

Current software versions are often transient. With the end of a research project or project financing occurs the risk that developed software components and systems are no longer maintained - meaning they will be outdated and inaccessible soon. This is either due to (a) security reasons, (b) incompatibility with new technologies (changing web standards, new operating systems standards, hardware incompatibility) or (c) dependencies to external systems (e.g. changing APIs). Whereas these issues - or the proposed solutions - are more technology-related, there are other areas of challenges, e.g. social, legal, or financial. The above described example of a visualisation framework provides an illustrative example of this context. The discontinuation of software environments may not pose a problem as long as publications of the research results are available. But as explained above this is not in the general interest of researchers, research funding and research institutions for various reasons such as transparency, reproducibility or refuse of research.

3.1. System security

Common problems are security issues in the software stack. These might be vulnerabilities in the operating system or software modules that are either third party or developed within the project. Some may be fixed with simple updates, which can be handled by the hosting institution, like updates of the operating system. But it is only a matter of time before the software is at the end of its support cycle or major upgrades might then threaten its stability or functionality. Separation is a relatively simple step for handling security issues. With access control and strict firewall rules on the network level, the system can be separated from other environments. The strongest way of

separation is complete isolation from other systems. Direct access is not possible in this case. Just a single gate- way service can access the system and communicate between the user and the system. But this limitation might cause problems: External dependencies, like the access to other databases, are limited. Another security layer of archived system states can be provided with the usage of templates to achieve safe states: A selected state of the research system will be transferred into a template. With each session, a new instance will be cloned from the template. This has the advantage, that every user will get a clean configuration in a defined state. However, in this scenario the state is fixed to a defined state: The session management and storage of data will become complex. System security poses a general challenge to the infrastructure provider as it is a fluid concept, advancing over time.

3.2. Compability

Software systems continuously have to adapt to new standards and technologies. Even if some technology stacks are relatively stable, their usage might change over time because of possible semantic drifts. There are basically two concepts to tackle hardware compatibility issues: either virtualisation or emulation (see also [van der Hoeven et.al. 2005]) of components. While emulation allows for running of operating systems and software on hardware they were not developed for, it costs extra computing power to emulate the needed hardware environment. A change of the already replaced hardware would cause additional work to reimplement the emulation. Virtualization has the clear benefit that the hardware appear as physical device to the system and the software modules. It comes with the disadvantage that the virtualised environment must be already capable directly on that hardware. But for most of the use cases considered here only standard hardware is used. Therefore the advantage of running more than one virtualised system and the relatively simple management of these system makes the virtualisation the state of the art approach.

3.3. Dependencies

With the increasing bandwidth and speed of data networks (including the internet) it be- came common to outsource data and software modules that are only (down-)loaded upon request. This approach of course enables compatibility and simplifies maintenance of widely used information. On the other hand it causes dependencies of a rapidly changing environment where the data provider has no influence on. Additional steps need to be taken in order to reduce these number as far as possible whereas the limitations are e.g. legal issues or data volume. On the one hand dependencies form a challenge for the infrastructure provider, on the hand the benefits for the research are obvious: it is possible to integrate very different types of data and content and in this way to enhance research. As a relativisation one has to take into account that the integration of external libraries, modules or applications is only possible with a certain level of standardisation. In this standardisation may also lay the key to the technological solution for archiving and referencing these kinds of data.

4. The citation of unambiguous system states via Fragment PIDs

The usage of PIDs is common to ensure stable references to publications and is increas- ingly accepted for file?based data to. The uniqueness and stability of PIDs has to be guar- anteed and realised by the PID service provider. These service providers use a specific PID system that may be distinguished by its functionality. E.g. the ePIC9 PID service offers the creation of Fragment PIDs. These are identifiers that not only can be resolved to a given location, but also allow to forward parameters when the PID is resolved. With this it becomes possible to make a stable reference to defined system conditions - but it also requires some effort on the side of the referenced system. The Fragment PID can be used to present the user predefined configurations of a system. This could for instance be a visualisation with specific parameters or a simulation of a specific state. As mentioned in the introduction PIDs are a common means to cite a very broad range of various objects. In the humanities PIDs are used to identify collections, content or objects. PIDs are not only able to reference to definite objects but may also reference object fragments with the usage of a Fragment PID. This may be passages of text or illustrations or links to certain sections in digital media by following examples:

- http://www.domain.org/book1@page=10
- http://www.domain.org/video1@begin=10&end=20

where in this example *book1* and *video1* represent the PID. The naming schema for such PIDs differs between the existing PID systems and is not subject of discussion in this paper. But important to note is that with those identifiers an unlimited number of fragments in an entity can be referenced and provide the level of granularity that is necessary for scientific citations.

5. The Humanities Data Centre as use case for referencing

The infrastructure for long-term storage and provision of research data in the humanities is only beginning to emerge, but is currently not as developed as in other domains such as astrophysics or climate research. Therefore, the Humanities Data Centre (HDC), aims to establish a data centre for research data from the humanities. The project will enter the construction phase to a later point. Therefore the described approach has a prototypical character. Above all looms the question of sustaining a research data centre, meaning that the proposed solutions not only have to meet the demand of the researchers but also have to address the conditions and resources available to the infrastructure provider (in terms of costs). As a consequence we wanted to identify suitable and already available solutions and components to compile the HDC service portfolio (c.f. Figure 2). For a de- tailed description of the HDC's initial service portfolio consult the project website. With regard to the question of referencing complex software environments (as representations of complex research data) and addressing the above mentioned general challenges of sustainability and referencing we introduce the application preservation as service component.

5.1. Application preservation

The initial service portfolio of the HDC can on the one hand be described as modular to address complex use cases, and contains on the other hand innovative components such as the application preservation. The application preservation can be used for complex forms of research data that fit into the above described pattern. It shall guarantee access to a stable version of a component (sustainability) - such as a virtual research environment, a complex database, or a visualisation framework - and enables the user to reference not only the component as a whole but also individual fixed states (referencing), e.g. to generate a certain query term or a specific visualisation set.

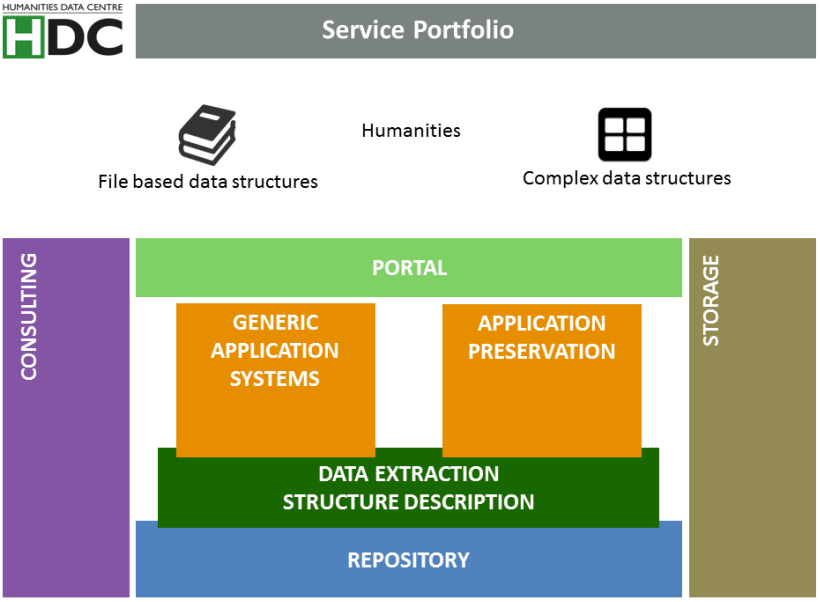

Figure 2. The HDC service portfolio that is currently under construction.

Data structures are preserved in their functional handover status and are only technically changed or maintained by the research data centre to extend its accessibility. Subject of the handover is ideally the whole data structure (e.g. client server structure, de- pendent libraries and applications). The service focusses on the presentation and reproducibility of research results and methods, not implicitly on the re-usability of the data. It is obvious that preserved applications can only be provided for a limited time by the research data centre for reasons of security gaps or outdating components. An ordinary archive case will utilise a combination of the above depicted services. Nevertheless the application preservation seems as an attractive service as it allows an ample presentation of research compared to archiving of raw data or documentations and for that reason demanding a solution for referencing.

5.2. Architecture of the application preservation

The architecture of the application preservation prototype as described below is the result of a process of raising requirements by the researchers and of evaluating already available components. Beside its nature as prototype it has also to be seen as a compromise which will improve and be further developed only in practical use. The application preservation allows for three ways to access the research result according to the security level. Directly after the project the software is up to date and safe and may be accessed directly (option 1 in Figure 3). When the software module is vulnerable the access will be restricted via an archived browser only (options 2 and 3 in Figure 3). Basically the application preservation consists of these layers:

1. A cloud infrastructure providing the storage and computing capability for the preserved application (its snapshots) and the application itself.

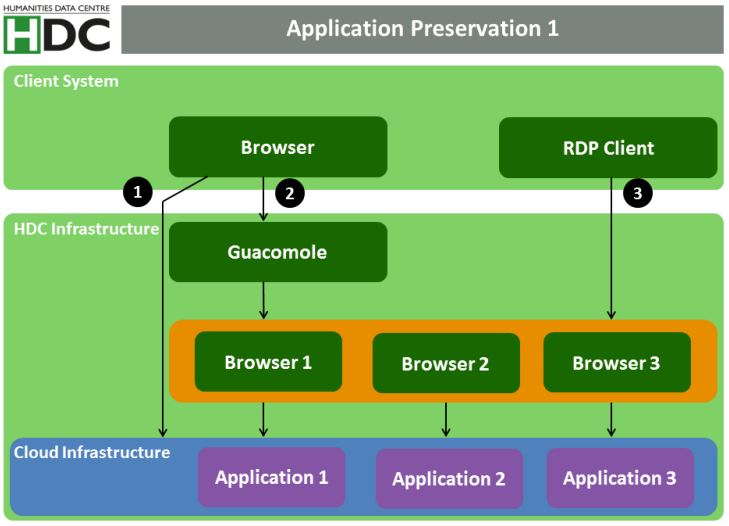

Figure 3. HDC infrastructure for application preservation. The numbered options depict different ways to access the preserved application.

2. A set of archived **browsers** of different brands and versions needed to visualise the application in the optimal way.

3. **Guacamole** [9, 10] serves as bridge between the user's environment - typically his browser - and the HDC infrastructure. It is able to handle RFP/VNC and RDP protocols to enable a remote access from the user to the preserved application without the inconvenience of installing additional software on the user site. A simple browser is sufficient to access the research data.

[9] http://www.pidconsortium.eu; last visited March 2016
[10] http://guac-dev.org

4. **PIDs** are used to reference the application or specific system state, e.g. a query term resulting in a specific data visualisation or database result.

The usage of PIDs in case of the application preservation can be divided in the following use cases:

1. A PID pointing directly to the URL of the visualisation (option 1 in Figure 3). This is the simplest way of referencing but the PID will become unresolvable quickly due to the reason of the application becomes outdated (c.f Sec. 3).

2. PID pointing at a specific connection configured in the guacamole client (option 2 in Figure 3). This allows a long term access to the application stored in a secure environment. The downside is an enhanced PID management effort to assure the reference to be stable.

3. In case of the third access method (option 3 in Figure 3) a PID usage is not possible due the a configuration of the RDP client on the user system. It still can be used to create a snapshot of an system state that later can be restored and referenced by a PID.

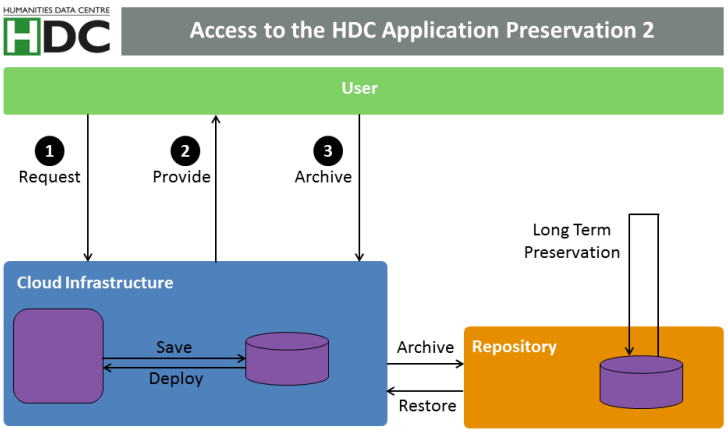

Figure 4. HDC cloud infrastructure for application preservation.

Also one need to distinguish between PIDs generated to reference the software model (the research result) or the HDC infrastructure. In the first case the software offers the service of citable states directly, e.g. by a URL search string. But if this functionality is missing the problem becomes more complex. The user queries or browser interactions have to be monitored and used to recover the desired state. Together with these informations the full state of the virtual environment has to be archived (c.f. Figure 4) which will then be referenced by PIDs that may also activate the system via the fragment mecha- nism. This functionality is only possible for option 2 in Figure 3.

6. Conclusion

This paper introduced a technological solution for the problem of referencing complex software environments. Complex software environments in this context serve as exem- plary representation for the emerging new forms of complex research data, not only in the humanities. A data visualisation framework has been introduced as an individual example for complex software environments. There are other classes of new research data but complex software environments are particularly suitable to demonstrate the challenges related to referencing these new forms of research data. There is a growing demand by researchers for referencing these forms of research data as they allow an ample view of a research project. The referencing of complex software environments could become prospectively a substantial element of the scientific impact and reputation of a researcher, therefore research data centres have to develop solutions of it. We continued with a description of the main characteristics of research data in the humanities, focussing on the more and more complex characteristics of data, which poses new requirements for ref- erencing compared to the long-established referencing of conventional types of content, e.g. text, and formats (monographs, journal articles). Following this, the main challenges in referencing complex forms of research data have been outlined: first ensuring the sus- tainability of actual software environments and second enabling to reference specific sys- tem states, such as a specific search string or query term in a database to ensure a certain granularity.

At this point it became clear that these new forms of research data pose crucial challenges for referencing and infrastructure providers in other, non-technological fields, e.g. legal issues. For instance the dependencies of complex software environments regard- ing external libraries or the licence status of raw data require coverage on side of the research data centre. A prototypical solution - the HDC application preservation - was introduced as a technological approach for referencing complex software environments. The approach mainly consists of a cloud infrastructure allowing remote access of users to preserved applications via an easy to use additional access layer. The problem of pointing to specific system states - such as a database query term or a specific visualisation data set - is solved via Fragment PIDs. Depending on the application these Fragment PIDs are used to forward URL extensions, resolve into connections to specific applications, or contain pointers that activate the complex software environment in exactly the state created as a snapshot by the editor of the reference. Although the paper remained on a technological level, there are challenges to be addressed reaching beyond the technological level such as legal, financial, interoperability or organisational questions. These

challenges also have to be addressed by a research data centre but are not be solved by one research data centre alone. Beyond such research data centres stand libraries and data centres, in the case of the HDC the Gottingen State and University Library (SUB) and the Gesellschaft für wissenschaftliche Datenverarbeitung (GWDG) as important providers of information infrastructure.

References

Sahle & Kronenwett 2013 P. Sahle & S. Kronenwett, Jenseits der Daten: U¨ berlegungen zu Datenzentren fu¨ r die Geisteswissenschaften am Beispiel des Ko¨ lner "Data Center for the Humanities"; Libreas 23 (2013); urn:nbn:de:kobv:11-100212726.

Kalman 2015 Tibor Kalman, "Fragment Identifiers, Template Handles". Presentation at the RDA-D Conference 2015.

Paskin 2010 Norman Paskin, "Digital Object Identifier (DOI) System" , Encyclopedia of Library and Information Sciences (3rd ed.), Taylor and Francis, pp. 1586-1592.

Aschenbrenner et.al. 2015 Andreas Aschenbrenner, Stefan Buddenbohm, Claudia Engelhardt, Ulrike Wut- tke: Humanities Data Centre - Angebote und Abla¨ ufe fu¨ r ein geisteswissenschaftliches Forschungsdatenzentrum; HDC-Projektbericht Nr. 1. 2015; http://humanities-data-centre.org/?page id=1036

van der Hoeven et.al. 2005 Jeffrey van der Hoeven, Hilde van Wijngaarden: Modular emulation as a long-term preservation strategy for digital objects. In: Lecture Notes in Computer Sci- ence. Volume 3652 2005. Research and Advanced Technology for Digital Libraries. 9th European Conference, ECDL 2005, Vienna, Austria, September 18-23, 2005. Proceedings. http://link.springer.com/chapter/10.1007%2F11551362_47