MMRepo - Storing qualitative and quantitative data into one big data repository

by Ingo Barkow[1] Catharina Wasner[2] Fabian Odoni[3]

Abstract

In recent years, the storage of qualitative data has been a challenge to data archives using repositories that are based on relational databases, as large files cannot really be represented well in these structures. Most of the time, two or more structures have to be in place e.g. a fileserver that includes versioning for large files and a relational database for the tabular information. These structures necessitate the handling of multiple systems at the same time. With the arrival of Hadoop and other big data technologies, qualitative data and quantitative data can now be stored as mixed mode data in the same structures. This paper will discuss our findings in developing an early prototype version of MMRepo at the University of Applied Sciences Eastern Switzerland HTW Chur. Our prototype of MMRepo is a combination of the Invenio portal solution from CERN with a Hadoop 2.0 cluster using the DDI 3.3 beta metadata scheme for data documentation.

Keywords

Mixed mode, qualitative, quantitative, big data, repository

Introduction

Storing different kinds of data from different domains and enhancing them with metadata has been the main workflow of research data centers, data archives or scientific repositories for many years. Nevertheless, the usage of different file formats, metadata standards or IT infrastructures is challenging for data managers and IT managers in those facilities. Mixed mode data – meaning data derived from qualitative research (e.g. open interview formats, ethnographic studies using video and audio) and quantitative research (e.g. questionnaires, cognitive tests) – which arise from the trend of combining different research designs (Punch, 2009), poses a particularly significant challenge. Data from qualitative research differ in size and structure very much from data derived from a quantitative design. This can be illustrated by the following example: An observation of a classroom full of students doing a computer-based test by filming high definition videos from multiple angles and recording different audio tracks. This is actually a mixed mode design as it combines a qualitative ethnographic study with a quantitative design (the computer-based test). The qualitative design will have as a result several gigabytes of video and audio files leading to processes like transcription for scientific processing. In contrast, the computer-based tests result in datasets that are often stored in formats for statistical packages (e.g. SPSS, Stata, SAS, R) and that contain variables, including variable and value labels. Both types of data will be documented with different metadata standards with hardly any overlap between them due to the difference in domains.

Storage of qualitative data and quantitative data in repositories

From an IT perspective, the interesting question is this: Can different research data types be stored within the same technical infrastructure (e.g. file servers, relational databases, data warehouses) to enable search functionalities for users within the same frontend across all different data types? The next chapter therefore looks at the current processes of storing these kind of data in selected repositories, gives examples and explanations on why different systems exist in the same organization, and explains the objectives and design considerations of a big data driven approach.

The current state of the art in storing mixed mode data

As mixed mode data vary considerably in size, documentation and type, storing quantitative data and qualitative data in one structure is a challenge. In most data archives, these are the most common ways to handle mixed mode data.

1.) Storing both types in a relational database

If a relational database is used as the data storage mechanism, the quantitative data can be ingested in its tabular format (e.g. by importing an Excel table or SPSS file into a database table). The associated metadata could be stored in database tables as well using table joins or referential integrity to connect metadata and data thus allowing for variable shopping baskets or personal extracts (see Amin et al., 2011). This means by using these features of some repository systems (e.g. Questasy from CentERdata) the user does not have to download a Scientific Use File (SUF) and clean it from unnecessary variables but can choose in the portal which variables should be exported from the system. In many of the organizations connected to IASSIST or the DDI community (e.g. GESIS, DIPF, IAB, CentERdata) storing of tabular data in relational databases is therefore still a preferred way to document and store quantitative research.

However, while a relational database is advantageous for strong quantitative data, it does not work well for qualitative data. Typically, qualitative data would be stored in a relational database as a binary large object (BLOB) or an object of similar type directly in the table, which increases table size dramatically. The database would then be linked to the content of a file server. Sometimes, hybrid technologies like file streams would be used (e.g. SQL Server 2014; see Mistry and Misner, 2014). All these technologies do not combine very well. Relational database systems are not very good at handling BLOBs. There are limitations in size per single cell (usually 2GB – see SQL Server BLOB varbinary(max) datatype; Mistry and Misner, 2014) and the inflation of size normally leads to performance issues as database servers are optimized for handling small atomic data like short strings or numbers. The external linkage between relational database and file server also does not work very well as the two systems are separated. If users perform changes on the file server, the database server uses the information where the files have been stored, usually leading to dead links. File streams as a hybrid technology try to avoid this problem by letting the database server handle the file server automatically. This technology is only available in enterprise database servers like SQL Server or Oracle. Unfortunately, file streams have some technical disadvantages like e.g. the data outside of the table will not be part of the backup environment of the database server anymore. Furthermore, the outsourced data on the external file server is not a part of a database transaction (meaning the database server will not be able to roll back a transaction in case of a technical problem or break-off from the user side). In summary, filestreams fix the problem of broken links between file servers and database servers, but do not include the features relational databases are known and used for.

2.) Storing all data as files on a file server

The other option would be to handle quantitative data and qualitative data in their file-based state on a file server. Metadata would be provided one of three ways: by an external relational database, be attached as attributes of files, or by adding additional files that contain the metadata information. While this is a good way to handle qualitative data, the advantages of processing structured tabular information of quantitative data within a relational database are lost. In particular, quantitative data is simply processed in its file form so users can download it, while advanced features like variable shopping basket, personal extracts, and search functionalities for variables or basic tabulation from a portal solution would be heavily limited.

Advantages of storing mixed mode research data in the same technical infrastructure

When talking about the advantages of storing mixed mode research data in one infrastructure first the question has to be answered why it can be problematic to have different storage structures for qualitative and quantitative data.

From an IT perspective two different kinds of repositories also mean handling two times a complete set of hardware and software infrastructure. This means the IT administration, IT support and software development have a much higher effort, as this can be completely separate systems (different servers, different operating systems, different repository software, different frontend). If the user is supposed to be provided with one portal solution to search through two or even more repositories (basing on different kinds of data) a meta search solution has to be provided. This means the user types in a search request and the meta search solution divides this into different search requests running on the different back ends, collecting the results and collating them into one common result. Consolidating a multitude of different systems is a huge effort and a simpler one stop solution in the backend would lead to huge advantages as only one system has to be catered from hardware and software side.

Examples from data centers

To get a clearer picture of how data repositories handle different kinds of data, two organizations from Germany were selected as examples – the German Institute for International Educational Research (DIPF)[4] and the Leibniz Institute for Social Sciences (GESIS)[5]. Both institutions archive and distribute research data and publications and run one or more research data centers accredited by the German Data Council (RatSWD)[6]

DIPF is currently running three different repositories for different purposes (see Bambey et. al. 2013 and Bambey et. al. 2012):

- Qualitative data (e.g. school observations in video) are stored in the Medienarchiv[7]
- Questionnaires and answer schemes are stored in the Database for Quality of Schools (DaQS)[8]
- Data documentation regarding the framework program for educational research in Germany (over 300 projects) is stored in the metadata database of the Verbund Forschungsdaten Bildung[9]

Those three separate repository systems are derived from former projects and are run by three different organizations – DIPF, GESIS and IQB.[10] From a technological point of view they are completely different and optimized for their respective content: qualitative data, quantitative data, and metadata on quantitative and qualitative data.

A similar approach can be seen at GESIS where the following structures can be found:

- Quantitative data, questionnaires and study documentation are stored in the Data Catalogue (DBK)[11]
- Variable-level information is stored in the ZACAT portal[12]
- Metadata on microdata are stored in the MISSY system[13]
- Full-text social science documents are stored in the Social Science Open Access Repository (SSOAR)[14]
- Historical studies and time series are stored in HISTAT[15]
- Time series data from social indicators are stored in the online information system SIMon[16]

The diversity of GESIS systems is due to independent developments in different departments but as well to the requirements of different target groups and diverse digital resources. An integrated search function across all sources is under development.[17]

The diversity of repository systems in DIPF and GESIS developed organically with the organizations over the years. This pattern can be seen in many similarly-sized institutions around the world. The development of integrated repository systems was constrained by the previously mentioned limitations of relational database and file server data storage systems. The separation of storage based on data types is therefore valid and has to be seen as a product of the times in which they were developed. Many of the repository systems have been running for several years or in some cases even decades.

Vision and objectives of a big data driven repository

Nevertheless, the question remains whether all research data can be unified in one system by using more modern approaches, not least because the IT administration of multiple existing systems alone takes up many resources. One candidate for this is big data technology. Possible benefits of using big data technology to store different research data types include

- The use of cluster based file systems. Big data file systems like HDFS2 automatically split files and workload across multiple servers including redundant copies. This means it has inbuilt fail save and processing capabilities already from the software side (no need to use expensive hardware).
- Access to robust semantic search systems like SOLR and Elasticsearch. As big data was originally developed for handling large unstructured data within search engines it comes with sophisticated search capabilities which go beyond a simple string matching, but also features understanding of underlying concepts in e.g. a full-text search to improve the results for the users.
- Applicability of text mining or natural language processing. As large quantities of unstructured data can be stored and processed in parallel big data offers the possibility to analyze these data with advanced syntactical or semantic methods.

Furthermore, big data technologies can manage unstructured data like qualitative data. Having all data types in one system would be less costly and resource intensive because no meta search platform as described before has to be set up and only one system has to be developed and maintained.

Another advantage from user perspective is new methods of data analytics and data science can be used e.g. for analyzing data across different data types with the possibilities like text mining or natural language processing offered within the big data solution. These can be combined with classical statistical analysis from the sphere of social sciences and offer a scientific value add.

Design considerations of using big data as a unified repository

Big data solutions like Hadoop[18] were originally developed as search engine companies like Yahoo or Google were not able to store the masses of data needed to offer their services. While relational databases or data warehouses rely heavily on clear data structures, the design paradigms of big data technology are different. Instead of having structured data on expensive cluster hardware, big data technology was designed to allow parallel processing of unstructured data on inexpensive hardware. This basis was extended over the years from a more file systems based approach like Hadoop 1.0 to a multi-layer platform as can be seen from figure 1.

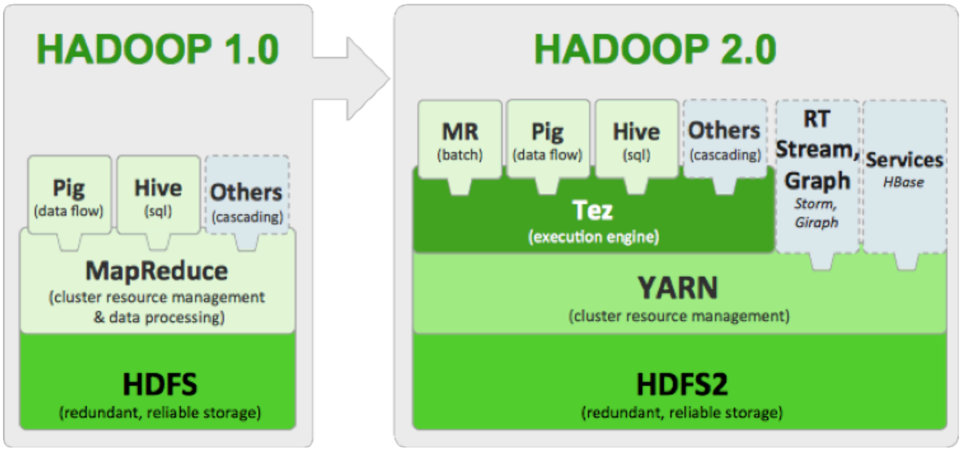
HADOOP 1.0
Pig (data flow)
Hive (sql)
Others (cascading)
MapReduce (cluster resource management & data processing)
HDFS (redundant, reliable storage)
HADOOP 2.0
MR (batch)
Pig (data flow)
Hive (sql)
Others (cascading)
RT Stream, Graph Storm, Giraph
Services HBase
Tez (execution engine)
YARN (cluster resource management)
HDFS2 (redundant, reliable storage)

Figure 1 – The development from Hadoop 1.0 to Hadoop 2.0 (Hortonworks 2013)

The addition of services is especially interesting in big data developments like Hadoop 2.0. One additional service in Hadoop 2.0 is Hbase, a non-relational (NoSQL) and column-oriented database modeled after Google's BigTable[19]. It runs on top of HDFS2, the native file system of Hadoop designed to store data among multiple computers, a cluster, by breaking up the data into blocks and distributing them throughout the cluster[20]. Another additional service is Hive which enables the use of SQL-like queries on the cluster.

Hadoop 2.0 could be the technical basis of a unified repository, whereby tabular content like metadata or qualitative data can be stored within Hive and large quantitative datafiles within HDFS2. This is the design idea which led to the MMRepo prototype project at the University of Applied Sciences Eastern Switzerland HTW Chur.

MMRepo as a prototype of a unified repository

As described in the chapter before, MMRepo was started as a prototype project to experiment with metadata, qualitative data and quantitative data within a single big data-based repository infrastructure. At the time of writing, MMRepo is a small project meant to test the performance and feasibility of the big data approach and must be considered a work in progress. The project started on January 1st, 2016 and its first phase ended on September 30th, 2016. The biggest obstacle in this respect was Invenio 3.0 final is as of today not released yet (March 2017). This means a large part of the final frontend testing had to be postponed to a later phase of the project. The current plan of Invenio specifies the release of the final version of 3.0 for summer 2017. The frontend testing will therefore be performed at a later date in the follow-up project called LifeCycleLab and not be part of this paper.

Structural design of MMRepo

The following test scenario has been set up to test the feasibility of the project. As the system's backend, a Hadoop 2.0 cluster is used with Hbase and HDFS2 as services. This backend will be combined with an Invenio[21] 3.0 beta frontend in the second project phase as the project is too small to develop portal functionalities by itself. The advantage of Invenio in this context is that it offers a modular framework for repositories where the data storage can be exchanged with something else (in our case a big data cluster). Furthermore, Invenio offers advanced features like semantic search or versioning, which benefit especially qualitative data. The structural design can be seen in the following figure. 2

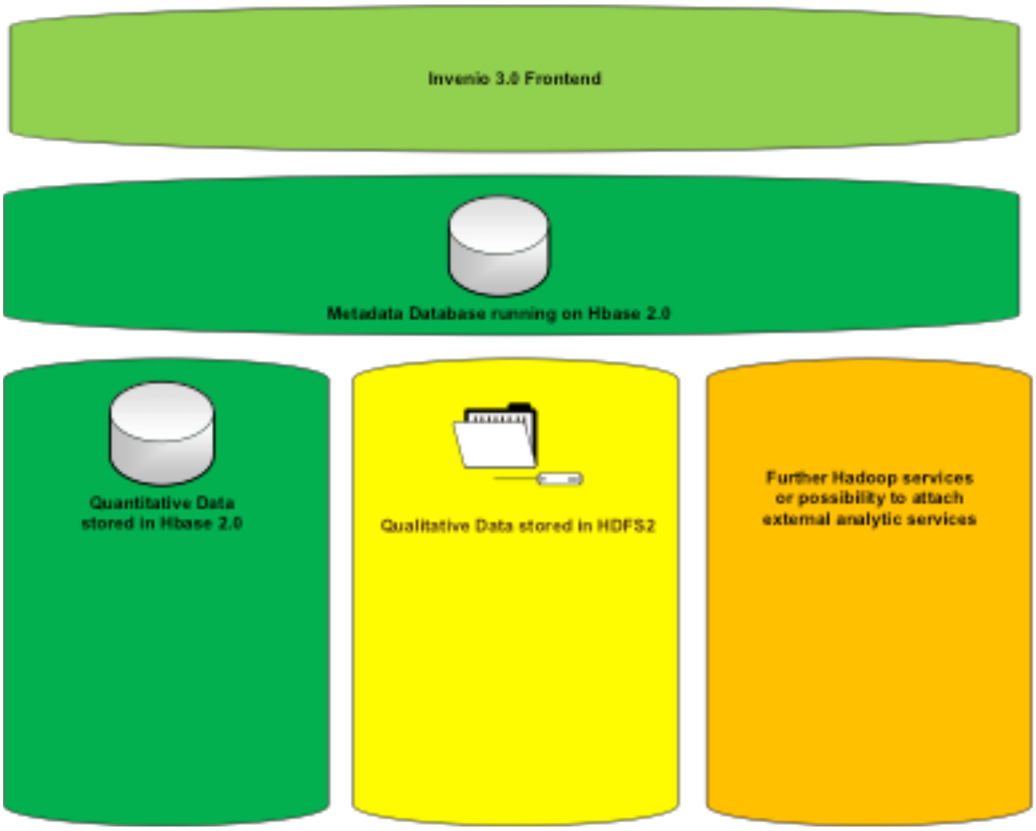
Invenio 3.0 Frontend
Metadata Database running on Hbase 2.0
Quantitative Data stored in Hbase 2.0
Qualitative Data stored in HDFS2
Further Hadoop services or possibility to attach external analytic services

Figure 2 – Software structure of MMRepo

The quantitative data and qualitative data used in the project are test data from previous studies at HTW Chur plus sample data from DIPF and the Research Data Center (FDZ) of the German Federal Employment Agency (BA) at the Institute for Employment Research (IAB)[22] . Therefore, a variety of test data has been imported into Hbase 2.0 and HDFS2. As the metadata schema for the quantitative data, Data Documentation Initiative (DDI)[23] Lifecycle 3.3 beta is used. For documenting the qualitative data, several internal groups at HTW are in favor of adopting the Metadata Encoding and Transmission Standard (METS)[24] . However, the final decision has not been made yet.

Hardware layout of the MMRepo prototype testing

As MMRepo uses Hadoop 2.0 as its backend, for proper testing, it is necessary to have a cluster environment. Only then, the advantages of parallel processing like the MapReduce algorithm can be exploited. Unfortunately, the project is much too small to employ even the least expensive servers. To simulate a multitude of nodes, the decision was made to set up the system on small third generation Raspberry Pi microcomputers.

Raspberry Pi 3 offers a quadcore 1.2 GHz ARM processor with 1GB of RAM plus SDHC card support and is an excellent less-costly solution for prototyping. A huge advantage for the Hadoop solution is additionally the support of LAN and Wireless LAN meaning two separated networks. For the current test setup the cluster network (LAN) and the user access (WLAN) can be separated as mixing the network traffic

Figure 3 – Raspberry Pi 3 (taken from www.raspberrypi.org)

from users and between server nodes might influence the results by slowing down reaction times.

From the hardware layout of big data solutions it does not make sense to test the capabilities by setting up e.g. two large powerful servers with inbuilt redudant hardware (e.g. multicore, multiple redudandant hard disks, fail safe networking) as the Hadoop software layout rather demands a high number of cheap servers with inexpensive hardware which can operate in parallel. The redudancy and the parallel processing is provided by the big data software itself. We also decided against employing virtualization (e.g. VMware vSphere, Citrix XenServer). If we tested the setup e.g. by setting up eight virtual servers in reality all virtual machines might end up on the same physical machine or might be shifted within several physical servers with different hardware layouts during the test (e.g. VMware – vMotion between nodes) thus influencing our results.

A prototype setup using the cheapest possible hardware is therefore closer to the specifications Hadoop was originally intended for. As we want to test primarily the search capabilities using an array of multiple servers we chose this setup for the prototype. In later phases of this project this prototype will be replaced by an array of inexpensive servers.

Results from the prototype

In late autumn 2016 we started to test the backend of the prototype after setting up the hardware with a limited number of Raspberry Pi nodes (one master, seven slaves) after it was clear Invenio 3.0 will not be released before the end of this project phase. We therefore decided on testing qualitative data and quantitative data on the backend of the cluster. Installing Hadoop on the inexpensive hardware was uneventful as there are in the meantime multiple How-Tos to be found (e.g. from IBM[25]). Some settings had to be modified manually in the environment variables, as Raspberry Pi is an unfamiliar hardware for Hadoop, but the How-Tos provided enough support.
To test the setup and set the correct block size for Hadoop the following datasets were implemented:

- US Census Data (2013_ACSSF_All_States_All_Tables)
  - Size: 469 MB
  - Format: Tables (*.csv)
  - Usage in Hadoop: Tables stored in Hbase (not relationally connected)

- Wikipedia – US Version (11/2016)
  - Size: 49 GB
  - Format: SQL dump
  - Usage in Hadoop: Relational database in Hbase

- Wikipedia – US Version (06/2008)
  - Size: 7.2 GB
  - Format: Static HTML dump
  - Usage in Hadoop: Files in Hbase

The selection of example datasets shows a mix of qualitative and quantitative data, but also the first problem in the prototypical setup. The amount of data used does not qualify as big data in a classical sense (4 Vs – Volume, Velocity, Veracity, Value – see e.g. Meyer, 2013) as the Volume is only several gigabytes while in big data repositories we rather aim at hundreds of terabytes or exabytes in the near future. Nevertheless, as the cluster is far from powerful from a systems' performance perspective the amount of data should be sufficient.

To see if the Hadoop cluster functions properly we started by implementing simple word count operations. The Raspberry Pi based cluster worked according to specifications, but ran very slowly. Part of the performance could be optimized by changing the block sizes. Also, the Hadoop software is not optimized to the Raspian operating system running underneath it, so the full performance of CPU and chipset is not used. As a first result it can be said the Raspberry setup is interesting to explore the possibilities of Hadoop on a real cluster especially for teaching purposes in an university environment, but it is not sufficient for performance testing or even productive setup.

From a conceptual perspective the results were much more promising. The ideas of storing files into the cluster-based filesystem HDFS and the tables into the NoSQL database Hbase worked fine. NoSQL database in this respect means "Not only SQL" a non-relational database layout which offers more flexibility in database layout while losing some transaction capabilities. The MySQL database dump from Wikipedia was imported into Hbase by using Sqoop. To see if the overall setup works we created search requests for HDFS2 and Hbase using SOLR which is embedded into Hadoop. As Hbase is built on top of HDFS2 as a column-oriented non-relational database system it worked well with SOLR so essentially we were able to use one hardware platform, one software platform and one search engine for the purpose of searching through qualitative and quantitative data as a first step.

Nevertheless, there is a limitation. Although SOLR searches through files and tables it does not mean the results are meaningful per se. In the end we get text extracts from documents and rows of tables as result sets which can be considered an intermediate step from a presentation point of view. A real frontend search solution needs adaptations in the presentation layer to have a user-friendly version of the results. Currently the whole setup from the backend to the portal is very crude and can be considered a proof of concept for further funding, but not an out-of-the-box usable solution. Nevertheless, the basic functionality is already available and therefore our project can continue. We are currently certain to be able to exchange our currently separated repositories into one big data solution, although the real development work on much better hardware has to start first.

References

Amin, A., Barkow, I., Kramer, S., Schiller, D. & Williams, J. (2011). Representing and Utilizing DDI in Relational Databases. Minnesota : DDI Working Paper Series [DOI:http://dx.doi.org/10.3886/DDIOtherTopics02].

Bambey, Doris; Rittberger, Marc (2013): Das Forschungsdatenzentrum (FDZ) Bildung des DIPF: Qualitative Daten der empirischen Bildungsforschung im Kontext. Standards und disziplinspezifische Lösungen. In: Huschka, Denis; Knoblauch, Hubert; Oellers, Claudia; Solga, Heike (2013) (Hrsg.): Forschungsinfrastrukturen für die Qualitative Sozialforschung. Berlin: Scivero-Verlag, S. 63-71. URL: http://ratswd.de/dl/downloads/forschungsinfrastrukturen_qualitative_ sozialforschung.pdf (20.03.2015).

Bambey, Doris; Reinhold, Anke; Rittberger, Marc (2012): Pädagogik und Erziehungswissenschaft. In: Neuroth, Heike; Strathmann, Stefan; Oßwald, Achim; Scheffel, Regine; Klump, Jens; Ludwig, Jens (Hrsg.): In: Langzeitarchivierung von Forschungsdaten. Eine Bestandsaufnahme. Boizenburg: Hülsbusch, S. 111-135.

Hortonworks (2013). Apache Hadoop Patterns of Use.

Andreas Meier (2013). Relationale und postrelationale Datenbanken – Leitfaden für die Praxis, Springer-Verlag.

Mistry, Ross and Stacia Misner (2014). Introducing Microsoft SQL Server 2014.

Punch, K. F. (2009). Introduction to research methods in education. London: Sage.

Notes

1. Ingo Barkow is an Associate Professor for Data Management at the University of Applied Sciences Eastern Switzerland HTW Chur and can be reached by email: ingo.barkow@htwchur.ch
2. Catharina Wasner is a research associate at the University of Applied Sciences Eastern Switzerland HTW Chur
3. Fabian Odoni is a research associate at the University of Applied Sciences Eastern Switzerland HTW Chur
4. http://www.dipf.de
5. http://www.gesis.org
6. http://www.ratswd.de
7. http://www.fachportal-paedagogik.de/forschungsdaten_bildung/medien.php?la=de
8. http://daqs.fachportal-paedagogik.de
9. http://www.forschungsdaten-bildung.de
10. https://www.iqb.hu-berlin.de
11. https://dbk.gesis.org
12. http://zacat.gesis.org
13. http://www.gesis.org/missy
14. http://www.ssoar.info
15. http://www.gesis.org/histat
16. http://gesis-simon.de
17. http://www.gesis.org/en/research/applied-computer-and-information-science/information-retrieval/
18. http://hadoop.apache.org/
19. http://www-01.ibm.com/software/data/infosphere/hadoop/hbase
20. http://www-01.ibm.com/software/data/infosphere/hadoop/hdfs

21 http://invenio-software.org

22 http://fdz.iab.de

23. http://www.ddialliance.org
24. http://www.loc.gov/standards/mets
25. Alan Verdugo - Building a Hadoop Cluster with Raspberry Pi - https://developer.ibm.com/recipes/tutorials/building-a-hadoop-cluster-with-raspberry-pi/#r_overview