10-12-2023

Implementation of a Federated Information System by Means of Reuse of Research Data Archived in Research Data Repositories

SYLVIA MELZER

STEFAN THIEMANN

SIMON SCHIFF

RALF MÖLLER

Implementation of a Federated Information System by Means of Reuse of Research Data Archived in Research Data Repositories

PRACTICE PAPER

SYLVIA MELZER
STEFAN THIEMANN
SIMON SCHIFF
RALF MÖLLER

*Author affiliations can be found in the back matter of this article

ubiquity press

ABSTRACT

At universities, research data is increasingly stored in research data repositories according to a data management plan (DMP) and thus made available for further use. The challenge of reusing hundreds, thousands, or millions of data sets is to obtain an overview of the data in a short period of time and to search through all the data. The high variability of the formats used to store research data requires a new approach to data reusability that focuses on the visualisation and searchability of archived research data, which can also be combined with each other. In this article, we present a practical DMP that describes how information systems can be created on demand by reusing research data archived in research data repositories and how these systems can be merged into a federated information system. As a result, in our projects, information systems have been created in minutes or a couple of hours with few resources. The initial effort to create a federated system remains; however, this allows federated searches to be performed. Extending a federated system to include other information systems can then be accomplished by making a few configurations and manageable adjustments to the source code.

CORRESPONDING AUTHOR:
Sylvia Melzer
Universität Hamburg, Centre for the Study of Manuscript Cultures, Warburgstraße 26, 20354 Hamburg, Germany
sylvia.melzer@uni-hamburg.de

KEYWORDS:
data management plan; databasing on demand; federated information system; FAIR principles

TO CITE THIS ARTICLE:
Melzer, S, Thiemann, S, Schiff, S and Möller, R. 2023. Implementation of a Federated Information System by Means of Reuse of Research Data Archived in Research Data Repositories. *Data Science Journal*, 22: 39, pp. 1–13. DOI: https://doi.org/10.5334/dsj-2023-039

1 INTRODUCTION

Data management plans (DMPs) and tools exist to support researchers in securing and archiving their research data according to a predefined, structured process. Researchers use different tools, have different workflows, and store their research data in various formats.

Sharing data can, then, become a major challenge if the data produced is not in the format needed by other researchers. A transformation of the data formats then becomes necessary. While there are some tools, such as eXtensible Stylesheet Language (XSLT) sheets (XSLT Working Group 2022) or Pandoc (MacFarlane 2022), that can transform data from one standardised format to another. In the field of humanities research, data is often found in project-specific formats, or the transformation is not loss-free. The formats chosen are widely used in the relevant community, but there is a lack of tools that can perform a conversion from one format to the other. The project 'Understanding Written Artefacts' (UWA) at the Universität Hamburg has therefore implemented a DMP to archive research data in a research data repository (RDR) wherever possible. They have already developed scripts that can transfer the data archived in the RDR into an information system on demand, providing researchers with a new way to work with archived data. An exchange across project boundaries is also made possible, as the scripts have been developed in such a way that they can be adapted to the needs of researchers with relatively little programming effort.

In this article, we describe the DMP of the Universität Hamburg in Section 3 and how this DMP is implemented in the following sections. Section 4 describes how research data is archived. Section 5 and Section 6 describe a time-saving and resource-efficient way to reuse research data from research data repositories transformed into an information system. Section 7 shows how to implement a federated information system that emerges from a network of created information systems. An application and the results are described in Section 8. In Section 9, we conclude our article contributions and give a short outlook for further work.

2 RELATED WORK

In a nutshell, it can be stated that there are many DMPs that aim to primarily archive research data over a long period of time and thus make it available to other researchers (cf. Harvard University 2022). Tools and RDRs have emerged that provide a structured description of the data and thus implement the requirements demanded by the funding programme or other regulations, without every researcher having to explicitly deal with the topic themselves. So far, RDRs are therefore ideally suited for archiving data.

Archiving research data over years creates unmanageable amounts of data, posing a new challenge when it comes to reuse. Approaches such as databasing on demand (DBoD) exist to transfer large amounts of data into information systems (Melzer, Schiff, et al. 2022). However, these tools have to be used separately. Integration of tools for reusing data in research data repositories is still missing. This article presents how the integration of DBoD in the RDR can be achieved. Additionally, it demonstrates how information systems can be created in a short period of time and with few IT resources, as well as how information systems can be merged into a federated information system to support federated search.

3 DATA MANAGEMENT PLAN

Several studies have shown the importance of a DMP in ensuring that research data are well-managed, organised, and preserved for future use (Thanos 2017). A DMP is a formal document. The document describes how to manage data during and after a research project. It describes additionally what type of data will be used for research, how it will be collected, organised, and stored, and what formats will be used (Harvard University 2022). Funding programmes such as the German Research Foundation (Deutsche Forschungsgemeinschaft; DFG) are currently calling for a DMP because the digital turn in science has benefited access to research data, methodological development in processing research data, and analytical methods for answering complex research questions. DFG (2021) lists the requirements for handling research data that must be described in the DMP. The requirements include:

- Data description: It must be described how new data is generated in a project, whether it can be reused, which data formats are used, and in what way and to what extent the data are used for further processing.
- Documentation and data quality: The procedures for describing the data in an understandable way and the tools used to ensure high data quality must be described.
- Storage and technical archiving: It must be described how the data will be stored and archived for at least 10 years during and after the project period. This also includes regulating access and user rights.
- Legal obligations and conditions: The legal particularities of handling research data must also be identified and regulated. The implications or limitations for subsequent publication or accessibility should also be assessed. For example, in the field of humanities, in some cases access to original artefacts is only granted if assurances are given that no images of them will be published.
- Data exchange and long-term data accessibility: A description must be given of how the records can be reused, where they are stored, how long they are stored, and when they are made available to third parties.
- Responsibilities and resources: It must be described who is responsible for the adequate handling of research data within a project and which resources are needed for that. This also includes describing who is responsible for curating the data after the project has ended.

Federal programmes such as the DFG or the European Commission also stipulate that research data should be openly accessible according to the FAIR ([F]indable, [A]ccessible, [I]nteroperable, [R]eusable) principles. That is, research data should be FAIR to increase research efficiency and transparency.

- Findable means that research data can be found on the internet via associated metadata and is citable via unique identifiers.
- Accessible means that research data are accessible openly or on request via standardised protocols.
- Interoperable means that research data can be technically reused through software and integrated with other data.
- Reusable means that research data are well documented and can be used for new research.

We therefore argue that the FAIR principles should also be anchored in the DMP.

The FAIR principles focus primarily on characteristics of data that facilitate increased data sharing between institutions. In the field of humanities, scholars often have to deal with historical artefacts, which may also be digital, from a wide variety of countries. The publication of these digitised artefacts can lead to problems from an ethical perspective, so we strongly recommend considering ethical issues in the DMP.

A RDR, other tools, and guidance were provided at the university to meet the research data management requirements required by the grant programme so that research data are effectively managed, shared, and utilised while adhering to the FAIR principles and ethical standards. These requirements include the following:

- RDR incl. manual for archiving research data (Universität Hamburg 2022).
- Network drive storage space for archiving sensitive research data.
- The database management tool Heurist (HEURIST 2022), including manual for building project-specific information systems.
- Guideline for ethical and responsible research.

In the following sections, we discuss how we have practically applied the DMP with a focus on data reusability.

4 ARCHIVING RESEARCH DATA

Research data management (RDM) is the overall process that guides researchers through the various phases of the data lifecycle and enables scientists and all other stakeholders to make the most of the research data they generate.

At the Center for Sustainable Research Data Management of the Universität Hamburg, the RDR was developed to make research data accessible and to archive them according to the FAIR principles, the DMP requirements mentioned in Section 3, and other regulations (Universität Hamburg 2022). That means that if researchers use the RDR, which has already implemented the FAIR principles, they do not have to worry about how to implement them themselves. RDR is similar to Zenodo (Research and OpenAIRE 2013) and both are based on Invenio. Zenodo is an open-access repository of research data and journal publications, represented according to the DataCite.org schema (Working Group 2021). DataCite.org provides persistent digital object identifiers (DOIs) for research data to assist researchers in locating, identifying, and citing research data. Research data stored in RDR include publications, measurement data, laboratory values, audiovisual information, texts, objects from collections or samples, interviews, and software. In brief, it can be stated that the RDR at the Universität Hamburg implements all the requirements of the DMP mentioned in Section 3 as follows:

- Open: The uploaded files can be made available in an open, restricted or closed manner, or be subject to a blocking period (embargo).
- Citable: Each entry in the repository is assigned a permanent DOI and can be found and cited through it.
- Secure: Data sets of up to 50 gigabytes can be uploaded via the web interface (more on request) and are stored securely.
- Sustainable: The retention period of the files is at least 10 years.
- Flexible: All materials that are in digital form are suitable for long-term storage in the repository. Any file format is possible.

Each file must be described when uploaded, and since in addition to the description of the research data, metadata must be included according to the Datacite.org schema, this ensures high quality data storage and also a uniform understanding of data collection and documentation.

At the Universität Hamburg, there is a range of guidance on how research data should be stored. Storing data in the RDR is just one of the other options used. Sensitive data, which may not be published for ethical reasons, for example, are stored on a server or kept in another way. While DataCite.org only aims to capture metadata, the RDR offers a description field that can be used to describe the research data in more detail. Since this description field corresponds to a text field, the entries can be easily searched via this field. Therefore, the Universität Hamburg offers the possibility to use the database management tool Heurist (HEURIST 2022) to create a database. This tool is particularly suitable for the field of humanities to store research data in a Heurist database instance during the duration of the project. However, if a researcher is interested in the reusability of the archived data, there is a big challenge if there are suddenly 20,000 DOCX or XML files and the researcher wants to get an overview of the content in a short time.

While human-readable formats such as DOCX, CSV, and PDF are somewhat easier to grasp visually, so-called RDR previewers are needed to display machine-readable formats such as Text Encoding Initiative (TEI) described in (Text Encoding Initiative 2020) and EpiDoc described in (Elliott et al. 2022) (both of which are special XML formats) according to the requirements of humanities scholars.

The Universität Hamburg has developed the RDR previewer for displaying content from CSV files (see Thiemann 2022) and a basis for building information systems on demand for some formats, e.g., TEI or EpiDoc, used in the projects to enable reusability of archived research data in a short time (see Melzer, Schiff et al. 2022; Schiff et al. 2022).

5 PRE-PROCESSING OF ARCHIVED RESEARCH DATA FOR DATABASING ON DEMAND

Research data can take many forms, including audio, video, images, and documents in various formats, as mentioned above. The latter can be structured, such as HyperText Markup Language (HTML), or unstructured, such as plain text files (TXTs). HTML is used to markup specific paragraphs in a TXT file, indicating elements like chapters, bold text, or anchor links that refer to other HTML pages accessible through a specific Uniform Resource Locator (URL). That is similar to Microsoft Word DOCX documents, which are ZIP files containing multiple extensible markup language (XML) documents. XML is used to highlight specific paragraphs in the DOCX document, similar to how HTML does it, and its schema is called office open XML (OOXML) (Dick 2020). For example, a small excerpt of a vocabulary index written in Microsoft Word DOCX is shown in Listing 1. Without any markup, it would be difficult to distinguish between a title, a chapter, and a textual paragraph. Line 1 in Listing 1 represents a title, Line 2 represents a chapter, and Line 3 represents a textual paragraph.

Listing 1 Excerpt of a vocabulary index.

```
1 Index 1
2 A
3 acai____DEDR 37: v. 4. to move - v.r. 40.5 77.13 96.5v 96.6 102.4
      ↪ 162.8 272.9 298.6 302.2 340.22
```

For differentiating between a title, a chapter and a textual paragraph, one needs to extract the XML files from the DOCX document. Among others, it contains a `document.xml` file, containing the actual textual content of the document at where textual paragraphs are marked up with XML tags. An excerpt of the corresponding `document.xml` with respect to the textual content of the Word DOCX document listed in Listing 1 is listed in Listing 2. The XML element `<w:pStyle w:val="Titel">` (German: Titel, Englisch: title) in Line 6 highlights the textual paragraph "Index 1" as being a title.

Listing 2 Excerpt of a document.xml.

```
1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <w:document ...>
3   <w:body>
4     <w:p ...>
5       <w:pPr>
6         <w:pStyle w:val="Titel"/>
7         ...
8       </w:pPr>
9       <w:r>
10        ...
11        <w:t>Index 1</w:t>
12      </w:r>
13    </w:p>
14    ...
15  </w:body>
16  ...
17 </w:document>
```

Other parts of the XML document listed in Listing 2 markup Line 2 in Listing 1 as being a chapter and Line 3 as a textual paragraph, where "acai" is separated by a tab from the rest of the text. Everything that is semantically different in Listing 1 can be either separated by the textual content only, such as a tab separating "acai" from the rest of the text or by the markup of the corresponding `document.xml`. Hence, we are able to transform the vocabulary index, written in Microsoft Word, automatically into a TEI document. The TEI format is more commonly used in the humanities for data storage and exchange.

First, we transform the document listed in Listing 1 into an intermediate one listed in Listing 3 where everything that is semantically differentiable is also syntactically differentiable.

```
<!TITLE!>Index 1
<!CHAPTER!>A
acai<!TAB!>DEDR 37: v. 4. to move - v.r. 40.5 77.13 96.5v 96.6
    ↪ 102.4 162.8 272.9 298.6 302.2 340.22
```

Listing 3 Transformed vocabulary index.

We use Antlr4 (Parr 2013) for specifying the controlled natural language in the intermediate document. Antlr4 is a tool for transforming a grammar and a lexer written in Backus–Naur form (BNF), a formal language description, into the source code of a programming language, such as Java, Python, or C++. During parsing, the parser executes application-specific code, which one can include into the grammar (Parr 2013). All languages, which can be formally described with a non-left-recursive context-free grammar, are supported (Parr, Harwell, and Fisher 2014) and the controlled language of the intermediate document listed in Listing 3 is one of them. The specification is automatically transformed into a Python program, which we modify to load the contents of the vocabulary index into a *PostgreSQL* database (PostgreSQL Global Development Group 2022). Using SQL (structured query language), the data loaded into the *PostgreSQL* can be transformed into any format at ease, such as but not limited to TEI as listed in Listing 4.

```
<?xml version="1.0" standalone="yes"?>
<TEI xmlns="http://www.tei-c.org/ns/1.0">
  <teiHeader>
    <fileDesc>
      <titleStmt>
        <title>Word Index</title>
        <author>Eva Wilden</author>
      </titleStmt>
      <publicationStmt>
        <p></p>
      </publicationStmt>
      <sourceDesc>
      </sourceDesc>
    </fileDesc>
  </teiHeader>
  <text>
    <body>
      <div>
      <head>A</head>
        <superEntry>
          <entry>
            <form>
              <orth>acai</orth>
            </form>
            <def>DEDR 37: v. 4. to move v.r. 40.5 77.13&#xB0; <ref
                ↪  target="links/96_5">96.5v</ref> <ref target="
                ↪ links/96_6">96.6</ref> <ref target="links/102_4
                ↪ ">102.4</ref> <ref target="links/162_8_5
                ↪ ">162.8</ref> <ref target="links/272_9">272.9</
                ↪ ref> <ref target="links/298_6">298.6</ref> <ref
                ↪ target="links/302_2">302.2</ref> <ref target="
                ↪ links/340_22">340.22</ref></def>
          </entry>
          ...
        </superEntry>
        ...
      </div>
    </body>
  </text>
</TEI>
```

Listing 4 Exported document as TEI.

We have not only transformed the Microsoft Word DOCX document into TEI, we have additionally linked all references to poems at where they refer to.

6 REUSE OF RESEARCH DATA VIA DATABASING ON DEMAND

We have developed a DBoD framework so that users can access, configure, and create customised project-specific information systems on demand with few resources and in a short time. We used the tool Heurist (HEURIST 2022), an open-source database management system with a web front-end that allows researchers without prior IT knowledge to develop data models, store data, search, and publish data on a website. Heurist was chosen to create Heurist database instances from hundreds or thousands of TEI files or other machine-readable formats on demand. The DBoD process consists of the following steps (see Figure 1):

1. We have written a Python program that transforms all TEI files into one Heurist XML (HML) file.
2. The HML file was imported into a Heurist database instance.
3. A web page was created with the Heurist web editor, which displays the data based on the scholars' requirements and thus represents an information system.

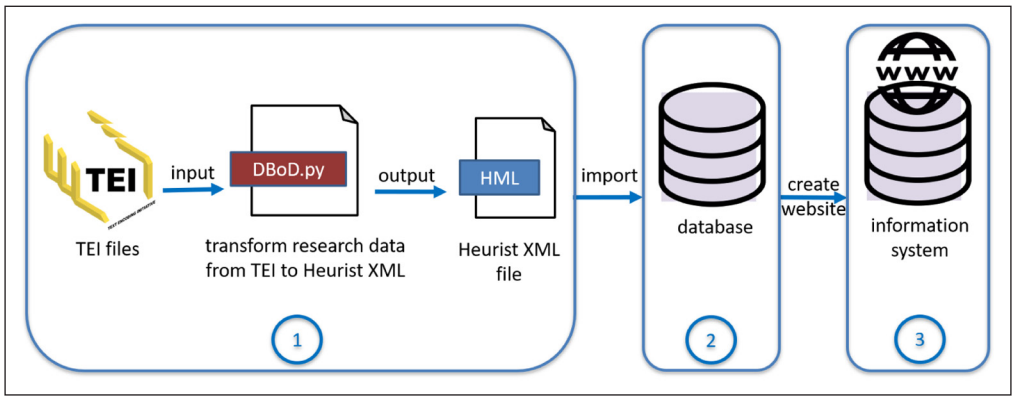

Figure 1 Databasing on demand: 1) transformation from all TEI files into one Heurist XML (HML) file, 2) importation of the HML file into a Heurist database instance, and 3) creation of an information system on top of the database instance.

A repository was created for the project NETamil, containing digital images of classical Tamil manuscripts on palm leaves and paper from Indian and European libraries, along with a descriptive catalogue, e-texts, critical editions, and annotated translations. The data was originally stored in a Word document and transformed into TEI (see Section 5). The automatic conversion of XML-encoded formats into a heuristic database instance has the feature of converting a large dataset into a new database instance within seconds, minutes, or hours instead of weeks or years.

In a prototype implementation, the DBoD process is integrated into a local version of the RDR so that researchers in the future can reuse data archived in the RDR with little effort for standardised or widely used formats. It is planned to transfer this process to the production version of the RDR. With DBoD, many information systems can be created on demand. Figure 2 represents the *NETamil2* Information System created on demand. On the left side is the search area, in the middle the result set, and on the right side, the project-specific data representation is displayed.

We have practically applied the DBoD process in the context of humanities projects. However, the approach is also applicable to other projects that have data in DOCX, JSON, or XML formats. Other databases can also be used to run the DBoD process. If other tools are used, however, it is necessary to ensure that the DMP is fulfilled.

Independent information systems have their limitations, and federated searches offer several advantages over them. Federated searches allow users to search across multiple databases and information systems simultaneously, which saves time and effort compared to searching each system individually. Federated searches also help to avoid redundancies in data and ensure that users have access to the most up-to-date and accurate information. This is particularly useful in interdisciplinary research, where data from multiple domains may be relevant to a single research question. Additionally, federated searches can facilitate collaboration between

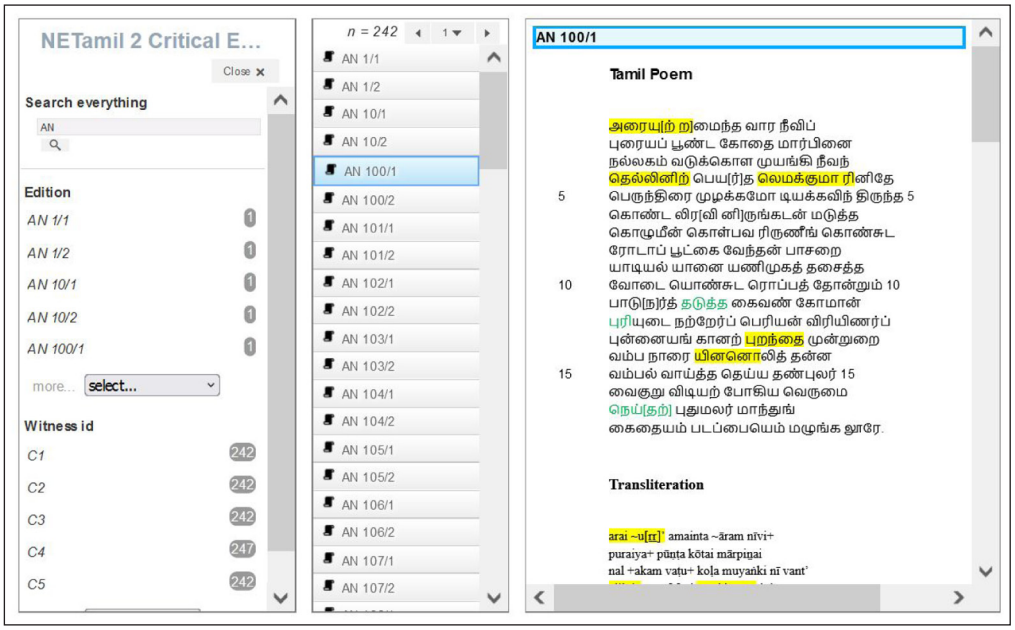

Figure 2 Website created with Heurist; left: search area, middle: result set, right: data representation.

researchers and institutions by enabling them to share data and resources more easily. Overall, federated searches provide a more integrated and streamlined approach to accessing and analysing data, which can lead to more efficient and effective research outcomes. The next section describes the process of creating links between the information systems built on demand, for example, to enable federated searches.

7 BUILDING A FEDERATED INFORMATION SYSTEM

A federated database system (FDBS) integrates multiple autonomous developed database systems into a single federated database. The constituent databases are interconnected via computer networks and may be physically decentralised. These databases can be connected via known interfaces. In Melzer, Thiemann, and Möller (2021); Melzer, Thiemann, Peukert, et al. (2022); and Melzer, Peukert, et al. (2022), the open-source message broker RabbitMQ is used to link the databases to an FDBS. In this approach, each database is equipped with its own RabbitMQ message broker. The distribution of multiple RabbitMQ brokers in different locations is called *broker federation*. Broker federation allows building messaging networks in which messages in one broker are automatically routed to another broker (Melzer, Thiemann, and Möller 2021).

RabbitMQ offers the *Advanced Messaging Queuing Protocol* (AMQP) as a standardised communication protocol. In the contributions Melzer, Thiemann, and Möller 2021; Melzer, Thiemann, Peukert, et al. 2022; Melzer, Peukert, et al. 2022 AMQP is used as a communication protocol in order to build an FDBS. AMQP defines the following three components:

1. The *message queue* stores messages that can be consumed by client applications.
2. The *exchange* receives messages from publisher applications and routes them to message queues.
3. The *binding* defines a relationship between a message queue and an exchange.

This allows the following communication paradigms to be implemented: send and receive, work queues, publish and subscribe, routing, topics, and request and reply. An implementation of the routing communication paradigm is performed, and the particular source code is presented in Figures 3 and 4. The Java-like scripting language BeanShell was chosen as the programming language. It is possible to choose other programming languages to write these publisher and client applications.

Each company or university has its database, the client applications for publishing and receiving messages to and from the RabbitMQ server, as well as a RabbitMQ broker with its individual broker configuration as input.

```
try
{
    ConnectionFactory factory = new ConnectionFactory();
    factory.setHost(host);
    factory.setVirtualHost(virtualHost);
    factory.setPort(port);
    factory.setUsername(userName);
    factory.setPassword(pw);
    Connection connection = factory.newConnection();
    Channel channel = connection.createChannel();

    try {
       channel.exchangeDeclare(EXCHANGE_NAME, "direct", true);
    } catch (IOException e1) {
       // TODO Auto-generated catch block
       e1.printStackTrace();
    }

    String severity = routingKey;

    channel.basicPublish(EXCHANGE_NAME, severity, null, message.getBytes("UTF-8"));

    channel.close();
    connection.close();
}
catch (TimeoutException e) {
    e.printStackTrace();
}
```

Figure 3 Source code for publishing messages.

As one example, the message queues *queueDb1*, *queueDb2*, and *queueDb3* are defined. They are in the same virtual host *dbFederation* (see Figure 5). The exchange is called *db.direct* (see Figure 6 above). The bindings with the particular routing key: queueDb1 → epiDoc, object, query; queueDb2 → object, queueDb3 → epiDoc, object are presented in Figure 6 (below). If a message is now published with the routing key epidoc, the queues *queueDb1* and *queueDb3*, which are connected to the routing key, receive the messages.

RabbitMQ has now been used to create a network between the RabbitMQ brokers. In order to receive data from the databases, another client application is needed to perform database queries (see Figure 7). In a workflow, the process can be described as follows:

1. The user enters a search query.
2. The search query is first forwarded to the user's database.
3. The answer to the query is sent to the user.
4. In parallel, the query is sent to RabbitMQ, where it is published.
5. The user receives further answers from other databases.

```
try {
    dok = channel.queueDeclare(QUEUE_NAME, true, false, false, null);
} catch (IOException e) {
    e.printStackTrace();
}
QueueingConsumer consumer = new QueueingConsumer(channel);
try {
    channel.basicConsume(QUEUE_NAME, true, consumer);
} catch (IOException e) {
    e.printStackTrace();
}
for(int i = 0; i<dok.getMessageCount(); i++){
    QueueingConsumer.Delivery delivery = null;
    try {
        delivery = consumer.nextDelivery();
    } catch (ShutdownSignalException e) {
        e.printStackTrace();
    } catch (ConsumerCancelledException e) {
        e.printStackTrace();
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
    reMessage = new String(delivery.getBody());
    print(" [x] Received '" + reMessage + "'");
    message = reMessage;
}
```

Figure 4 Source code for receiving messages.

Overview				Messages		
▽ **Virtual host**	**Name**	Features	**State**	**Ready**	**Unacked**	**Total**
dbFederation	**queueDb3**	D Args	idle	0	0	0
dbFederation	**queueDb2**	D Args	idle	0	0	0
dbFederation	**queueDb1**	D Args	idle	0	0	0

Figure 5 Defined queues on a RabbitMQ server.

It should be noted here that the answers may be correct, incorrect, or incomplete, and it is recommended to use appropriate schema matching procedures.

8 APPLICATION AND RESULTS

During the project NETamil at the Universität Hamburg, so-called critical editions of Tamil poems and a dictionary were represented in Word files. The transformation from Word to the TEI format was processed as described in Section 5. The information system for this project was built in two and a half hours via DBoD. (XSLT stylesheets were used in the transformation from TEI to HML to correctly display the content. These stylesheets caused the process to slow down. In other projects, the DBoD could be executed four-fifths of the time faster without using the stylesheets.) In this way, several information systems were built.

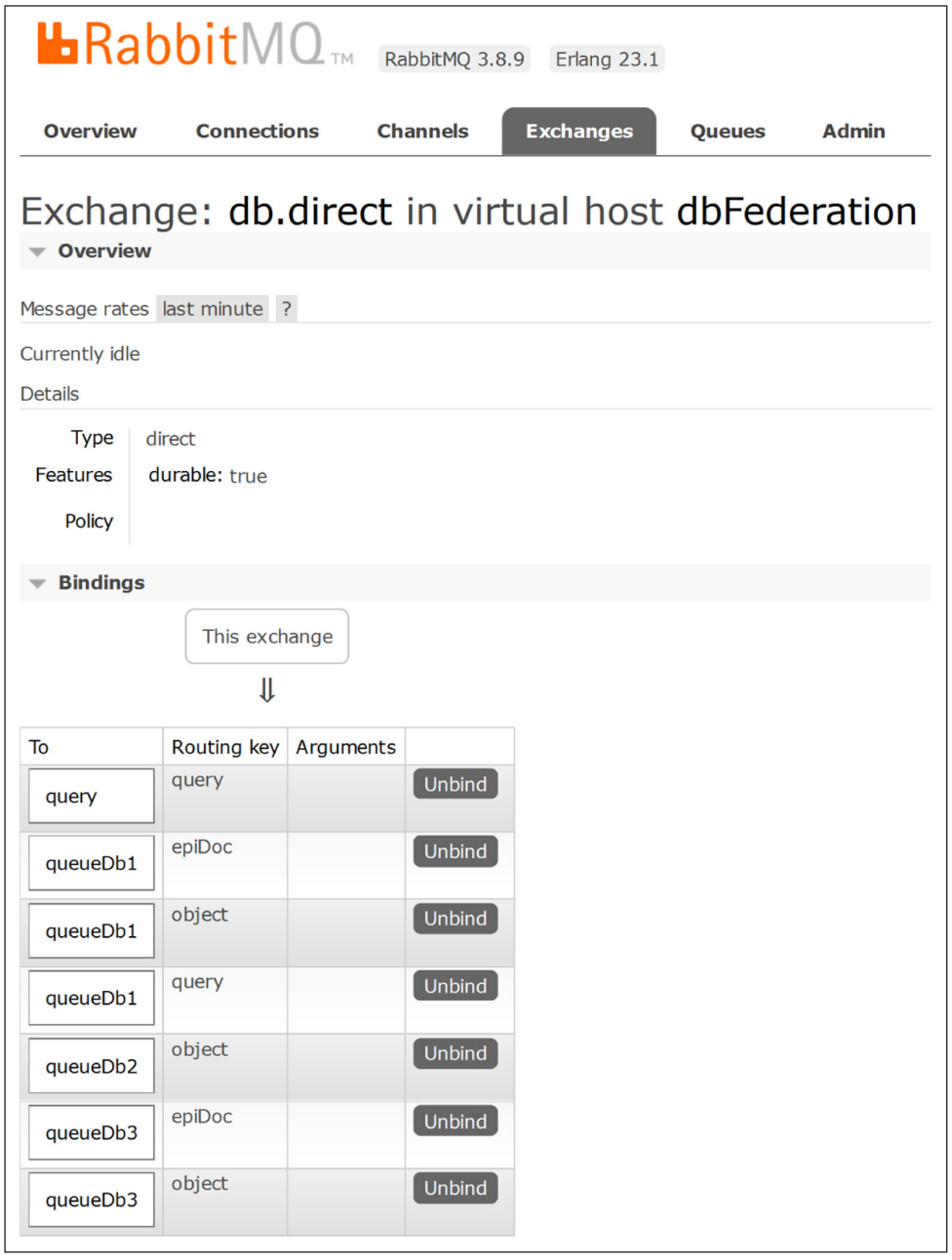

Figure 6 Defined bindings.

To link all the generated information systems, we used a notebook where a RabbitMQ server (version 3.8.9) and *MariaDB* (10.5.6) were installed. We created the databases *NETamil* using *MariaDB*. On a Raspberry Pi 4, we also installed a RabbitMQ server and *MariaDB*, where the database *NETamil2* was created. The database *NETamil2* has overlapping and different entries compared to *NETamil*. Both RabbitMQ servers were configured with specific message queues, exchanges, and bindings. As a result, an FDBS was implemented. In our example, queries from *NETamil* are answered using both the *NETamil* and the *NETamil2* databases. The answers returned by the databases sometimes showed incompleteness. This was because, for example, the date was displayed differently in the two databases. The date differed in language and presentation (given as a century or year). When the query is filtered by year, one of the two databases returns an empty result set as a response. In this case, the information integration problem must be solved. Nevertheless, federated queries can be advantageous in data analysis because redundancies of database entries can be avoided, and a dataset can be evaluated as a unit.

```
29      Statement st = null;
30
31      try {
32          st = conn.createStatement();
33      } catch (SQLException e) {
34          e.printStackTrace();
35      }
36      ResultSet rs = null;
37      try {
38          rs = st.executeQuery(query);
39      } catch (SQLException e) {
40          // TODO Auto-generated catch block
41         e.printStackTrace();
42      }
43      try {
44          ResultSetMetaData rsmd = rs.getMetaData();
45          int columnsNumber = rsmd.getColumnCount();
46
47          while (rs.next()) {
48          String printStr = "";
49            for(int i = 1; i <= columnsNumber; i++) {
50               printStr += rs.getString(i) + " ";
51            }
52            print(printStr);
53            result = printStr;
54          }
55      } catch (SQLException e) {
```

Figure 7 Source code for sending and receiving data from database.

9 CONCLUSION AND OUTLOOK

In this contribution, we have presented how research data stored in RDRs can be reused in other projects with low IT resources. We have shown that hundreds, thousands, or millions of archived data can be reused by allowing information systems to be built on demand via DBoD in a short period of time with the possibility to search through all the data. We have successfully tested the DBoD approach in several humanities projects, demonstrating its algorithmic validity and potential for implementation in other domains. The result we have achieved is a significant reduction in the implementation time of building domain-specific information systems for scholars without the help of an IT expert. In addition, we have shown that these information systems can be linked via RabbitMQ to enable federated searches. This approach meets the requirements of federal programmes, the implementation of which was recorded in a DMP through the tools and manuals used. Furthermore, the FAIR principles and ethical aspects are

automatically considered while reusing the data and building a federated information system on demand by means of reusing research data archived in the RDR. In the RDR, the research data has been archived in accordance with the DMP so that further processing can be assumed to be FAIR and ethical. In over five projects, information systems have been created in minutes to a couple of hours with few resources. The initial effort to create a federated system remains; however, this allows federated searches to be performed. Extending a federated system to include other information systems can then be accomplished by making a few configurations and manageable adjustments to the source code. Our goal is to extend the prototype implementation with external databases and then implement the DBoD process in the RDR.

FUNDING INFORMATION

This research was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy—EXC 2176 'Understanding Written Artefacts: Material, Interaction and Transmission in Manuscript Cultures', project no. 390893796.

COMPETING INTERESTS

The authors have no competing interests to declare.

AUTHOR AFFILIATIONS

Sylvia Melzer orcid.org/0000-0002-0144-5429
Universität Hamburg, Centre for the Study of Manuscript Cultures, Warburgstraße 26, 20354 Hamburg, Germany

Stefan Thiemann orcid.org/0000-0001-8300-2519
Universität Hamburg, Centre for the Study of Manuscript Cultures, Warburgstraße 26, 20354 Hamburg, Germany

Simon Schiff orcid.org/0000-0002-1986-3119
University of Lübeck, Institute of Information Systems Ratzeburger Allee 160, 23562 Lübeck, Germany

Ralf Möller orcid.org/0000-0002-1174-3323
University of Lübeck, Institute of Information Systems Ratzeburger Allee 160, 23562 Lübeck, Germany

REFERENCES

DFG. 2021. *Handling of research data*. Available at https://www.dfg.de/download/pdf/foerderung/grundlagen_dfg_foerderung/forschungsdaten/forschungsdaten_checkliste_en.pdf [Last accessed 12 December 2022].

Dick, D. 2020. *Office open XML*. Available at http://officeopenxml.com/anatomyofOOXML.php [Last accessed on 4 November 2022].

Elliott, T, et al. 2022. *EpiDoc guidelines: Ancient documents in TEI XML (Version 9)*. Available at https://epidoc.stoa.org/gl/latest/ [Last accessed 22 January 2022].

Harvard University. 2022. *Research data management @Harvard*. Available at https://researchdatamanagement .harvard.edu/data-management-plans [Last accessed 12 December 2022].

HEURIST. 2022. *A unique solution to the data management needs of humanities researchers.* Available at https://heuristnetwork.org/ [Last accessed 17 January 2022].

MacFarlane, J. 2022. *Pandoc user's guide.* Available at https://pandoc.org/MANUAL.html [Last accessed 4 October 2022].

Melzer, S, Peukert, H, et al. 2022. Model-based development of a federated database infrastructure to support the usability of cross-domain information systems. In: *2022 IEEE International Systems Conference (SysCon)* (IEEE SysCon 2022), Montreal, Canada, on 25–28 April 2022, pp. 1–8. DOI: https://doi.org/10.1109/SysCon53536.2022.9773811

Melzer, S, Schiff, S, et al. 2022. "Databasing on demand for research data repositories explained with a large epidoc dataset." In: *CENTERIS*.

Melzer, S, Thiemann, S and **Möller, R.** 2021. Modeling and simulating federated databases for early validation of federated searches using the broker-based SysML Toolbox". In: *IEEE International Systems Conference* (SysCon 2021), Vancouver, BC, Canada, April 15–May 15, 2021, pp. 1–6. DOI: https://doi.org/10.1109/SysCon48628.2021.9447055

Melzer, S, Thiemann, S, Peukert, H, et al. 2022. "Towards a model-based and variant-oriented development of a system of systems." *Advances in Science, Technology and Engineering Systems Journal*, 7(3): 19–31. DOI: https://doi.org/10.25046/aj070303

Parr, T. 2013. *The definitive ANTLR 4 reference*. Pragmatic Bookshelf, pp. 1–326.

Parr, T, Harwell, S and **Fisher, K.** 2014. Adaptive LL(*) parsing: The power of dynamic analysis. In: *ACM SIGPLAN Notices*, 49(10): 579–598. DOI: https://doi.org/10.1145/2714064.2660202

PostgreSQL Global Development Group. 2022. *PostgreSQL 15.0 documentation*. Available at https://www.postgresql.org/docs/15/index.html [Last accessed 12 September 2022]. DOI: https://doi.org/10.25495/7GXK-RD71

Research, European Organization for Nuclear and OpenAIRE. 2013. *Zenodo.en*. Available at https://www.zenodo.org/ [Last accessed 10 October 2022].

Schiff, S, et al. May 2022. TEI-based interactive critical editions. In: *15th IAPR International Workshop on Document Analysis Systems*. Lecture Notes in Computer Science (LNCS). Springer. DOI: https://doi.org/10.1007/978-3-031-06555-2_16

Text Encoding Initiative. 2020. *P5: Guidelines for electronic text encoding and interchange, version 4.0.0.* Last updated 13 February 2020, revision ccd19b0ba. Available at https://tei-c.org/Vault/P5/4.0.0/doc/tei-p5-doc/en/html/ [Last accessed 29 June 2022].

Thanos, C. 2017. Research data reusability: conceptual foundations, barriers and enabling technologies. *Publications*, 5(1): 2. DOI: https://doi.org/10.3390/publications5010002

Thiemann, S. 2022. *RDR previewer*. Available at https://tools.fdm.uni-hamburg.de/EDAK/sorttable.html [Last accessed 12 December 2022].

Universität Hamburg. 2022. *Research data repository*. Available at https://www.fdr.uni-hamburg.de/ [Last accessed 12 December 2022].

Working Group, DataCite Metadata. 2021. *DataCite metadata schema documentation for the publication and citation of research data and other research outputs*. Version 4.4. DataCite e.V. DOI: https://doi.org/10.14454/3w3z-sa82

XSLT Working Group. 2022. *XSLT cover page*. Available at https://www.w3.org/TR/xslt/ [Last accessed 4 October 2022].

TO CITE THIS ARTICLE:
Melzer, S, Thiemann, S, Schiff, S and Möller, R. 2023. Implementation of a Federated Information System by Means of Reuse of Research Data Archived in Research Data Repositories. *Data Science Journal*, 22: 39, pp. 1–13. DOI: https://doi.org/10.5334/dsj-2023-039

Submitted: 15 December 2022
Accepted: 22 August 2023
Published: 12 October 2023

Data Science Journal is a peer-reviewed open access journal published by Ubiquity Press.