A Metadata-Driven Approach to Panel Data Management and its Application in DDI on Rails

Dissertation

presented to the Faculty for Social Sciences, Economics, and
Business Administration at the University of Bamberg
in Partial Fulfillment of the Requirements for the Degree of

Doctor Rerum Politicarum

by
Marcel Hebing, Dipl. Soz.
born 4 August 1982 in Munich

Date of submission
18 November 2015

Advisors:
Prof. Dr. Susanne Rässler, University of Bamberg
Prof. Dr. Klaus Tochtermann, University of Kiel
Prof. Dr. Silke Anger, University of Bamberg

Declaration of academic honesty

I hereby confirm that my dissertation is the result of my own work. I did not receive any help or support from commercial consultants. All sources or materials applied are listed and specified in the dissertation. I have explicitly marked all material which has been quoted either literally or by content from the used sources. Furthermore, I confirm hat this dissertation has not yet been submitted as part of another examination process neither in identical or similar form.

Marcel Hebing

Bamberg, 18 November 2015

Abstract

This dissertation designs a metadata-driven infrastructure for panel data that aims to increase both the quality and the usability of the resulting research data. Data quality determines whether the data appropriately represent a particular aspect of our reality. Usability originates notably from a conceivable documentation, accessibility of the data, and interoperability with tools and other data sources. In a metadata-driven infrastructure, metadata are prepared before the digital objects and process steps that they describe. This enables data providers to utilize metadata for many purposes, including process control and data validation. Furthermore, a metadata-driven design reduces the overall costs of data production and facilitates the reuse of both data and metadata.

The main use case is the German Socio-Economic Panel (SOEP), but the results claim to be re-usable for other panel studies. The introduction of the Generic Longitudinal Business Process Model (GLBPM) and a general discussion of digital objects managed by panel studies provide a generic framework for the development of a metadata-driven infrastructure for panel studies. A first theoretical application presents two designs for variable linkage to support record linkage and statistical matching with structured metadata: concepts for omnidirectional relations and process models for unidirectional relations. Furthermore, a reference architecture for a metadata-driven infrastructure is designed and implemented. This provides a proof of concept for the previous discussion and an environment for the development of DDI on Rails. DDI on Rails is a data portal, optimized for the documentation and dissemination of panel data. The design considers the process model of the GLBPM, the generic discussion of digital objects, the design of a metadata-driven infrastructure, and the proposed solutions for variable linkage.

Contents

Introduction

A significant amount of scientific work in the social and economic sciences is based on secondary data—data that are not collected by the researcher conducting the analysis. When using secondary data, researchers depend on the data producers regarding two aspects of the data: quality and usability. Data quality determines whether the data appropriately represent a particular aspect of our reality. Usability originates notably from a conceivable documentation, accessibility of the data, and interoperability with tools and other data sources.

In this context, researchers usually assume that metadata are just one part of the documentation. However, metadata can support all aspects of data quality and usability. We can use them to validate research data, improving the quality. Data portals and other documentation systems can increase the accessibility of research data based on descriptive metadata. Furthermore, standardized or harmonized metadata can help researchers who are looking for suitable data sources that can be combined using record linkage or statistical matching—or even combine those data automatically.

The main claim of this dissertation is that metadata can be used for much more than just the retrospective documentation of data. To optimize the utilization of metadata, however, we have to rethink the way we manage and understand them. It is not sufficient to prepare a subset of metadata eventually, long after the actual research data have been collected and published. In a metadata-driven infrastructure, as proposed in chapter 4, metadata are prepared even before the object or task that they describe. This enables new use cases for metadata—for example, to control significant parts of the data management process or to validate the content of datasets and other digital objects. Furthermore, metadata are usually of much better quality when they are prepared at the same time as the respective object which they describe—and not months or years later.

The Data Documentation Initiative (DDI) and other metadata communities have changed their perspective on scientific metadata since the 1990s. The DDI Lifecycle model, for example, illustrates the idea of reusing metadata in subsequent iterations of a research project [1]. Furthermore, there is first work on metadata-driven approaches [2, 3, 4]. The existing literature, however, stays vague on what 'metadata-driven' means—the term is mostly used as a placeholder for all kinds of process designs that involve metadata. A major concern of this dissertation is to provide a generic but still precise understanding of the term 'metadata-driven'.

The context for the following discussion are panel studies that stand out because they have an iterative design by definition. The seven chapters start with a concrete example, the German Socio-Economic Panel (SOEP), but then abstract from it to bring the discussion on a rather generic level, based on the Generic Longitudinal Business Process Model (GLBPM) and a general discussion of digital objects. This provides the context for the actual introduction of a metadata-driven infrastructure and the concrete example of variable linkage. The first five chapters discuss individual research questions which are intended as complementing contributions to the main concern, table 1 provides an overview. The last two chapters apply the idea of a metadata-driven infrastructure to the design of a reference architecture and the development of the data portal DDI on Rails.

Position in science and related work

This dissertation is intended as a contribution to scientific data and metadata management in the field of social, economic, and behavioural sciences. The discussion of data and metadata, however, involves other disciplines—most notably, statistics and computer science. These domains are also represented by the three supervisors of this dissertation: Prof. Dr. Susanne Rässler (statistics and econometrics), Prof. Dr. Klaus Tochtermann (computer science), and Prof. Dr. Silke Anger (economics). Valuable inputs come from various research areas, including statistical matching and record linkage [e.g., 5, 6, 7, 8], data sharing and replication [e.g., 9, 10, 11, 12], scientific computing [e.g., 13, 14, 15], the metadata communities [e.g., 2, 16, 17, 18] and, in particular, the Data Documentation Initiative [e.g., 19, 20, 21, 22], panel studies [e.g., 23, 24, 25], software design [e.g., 26, 27, 28, 29], and the linked open data community [e.g., 30, 31, 32].

The idea for the development of DDI on Rails and this dissertation originated from my work as a software developer and data scientist at the German Socio-

Table 1: List of research questions and goals

	Research question
Chapter 1	Who are the users and producers of the SOEP data and what are their requirements regarding usability, data quality, and software support?
Chapter 2	What does a generic process model for panel studies look like and which digital objects can be identified in the model?
Chapter 3	Can the digital objects, identified in chapter 2, be modelled in a generic way; that is, can it be independent from the specific implementations in software tools and panel studies?
Chapter 4	How can the concept of a metadata-driven infrastructure optimize the production of panel data and increase the quality and usability of the resulting data?
Chapter 5	Which information about related variables do survey researchers require to analyse distributed data sources (e. g., statistical matching or record linkage) and how can they be covered in the metadata?
Chapter 6	Application 1: Introduction of a reference architecture
Chapter 7	Application 2: Introduction of DDI on Rails

Economic Panel (SOEP, located at the German Institute for Economic Research, DIW Berlin). It was accompanied with related research projects, including the work on the next version of the DDI standard (Data Documentation Initiative) [33], analysing the motivations and barriers for data sharing in academia as part of the Leibniz Research Alliance Science 2.0 [34, 35], linking online and panel data in the DFG-project Processes of Mate Choice in Online-Dating (PPOK) [36], the development of the Generic Longitudinal Business Process Model (GLBPM) [37], the establishment of the SOEP user surveys as a longitudinal study [38], and the development of research tools like the R package r2ddi [39]. The main project, however, was the development of the data portal DDI on Rails [40], a web-based application for the discovery and dissemination of panel data. I very much like to thank all partners in these projects for the insights they provided and the support I got during that time.

The development of DDI on Rails started with the intention to build a successor for SOEPinfo, which is the online documentation system for the SOEP data. While the former system was optimized for one study only (the SOEP), DDI on Rails is designed to be study-independent. The use cases in chapter 7 describe how the system is used to document the SOEP Core study, SOEP-related studies, and also external studies on paneldata.org [41]. The core functionality of both SOEPinfo and DDI on Rails enables researchers to search and discover variables and questions, including relationships amongst them. In the context of panel data, it is of particular interest to link related variables and questions over time to enable researchers to analyse the specific design of a panel study.

Confronting the theoretical discussion with the implementation of DDI on Rails and paneldata.org, the dissertation ensures that the theoretical results and the proposed infrastructure design are feasible in production for both sofware development and panel studies. At the same time, the discussion abstracts from the various examples and use cases and designs a generic framework for a metadata-driven infrastructure for panel studies. This ensures that the results are re-usable for panel studies around the world. These two aspects—(1) the confrontation of the theoretical results with actual software implementations and production systems; and (2) the generic approach in the discussion—are considered to be the core characteristics of this dissertation providing a unique contribution to the field of data management in the social, economic and behavioural sciences.

Outline

The dissertation consists of two parts. The first part designs a generic framework for the management and documentation of panel data, while also proposing workflows and data structures for the implementation of a metadata-driven infrastructure for panel data. The second part assesses the feasibility of the results from the first part by applying them in the design of a reference architecture and the development of DDI on Rails.

The main use case is the German Socio-Economic Panel (SOEP). The first chapter takes a closer look at the users and producers of the SOEP data, the process and structure of data production, and the needs of the researchers working with the data, asking: Who are the users and producers of the SOEP data and what are their requirements regarding usability, data quality, and software support? The detailed discussion of the SOEP provides a specific introduction into the field of panel studies, data production, and data re-use.

The dissertation and the development of DDI on Rails aim to present reusable solutions for panel studies. Focusing on the SOEP as the sole use case would be very unlikely to produce reusable results. In contrast, it would be overwhelming to take all possible use cases into account. The second chapter steers a middle course and presents a generic model for panel studies, the Generic Longitudinal Business Process Model (GLBPM) [37]. The original design of the GLBPM, however, lacks details on digital objects which accompany the production of panel data and provide the context for improvement of the overall process—a gap that the second chapter also aims to close, asking: What does a generic process model for panel studies look like and which digital objects can be identified in the model?

The second chapter identifies five classes of digital objects that are of particular interest for optimizing the production and documentation of panel data: study descriptions, questionnaires, research data, transformation scripts, and the documentation. For each digital object, various technical implementations and data models are used in production—again, we have to abstract from the actual implementations to facilitate reusable solutions. This, however, is not possible for all digital objects, as some of the implementations, in particular for transformation scripts, are conflicting. Thus, chapter 3 asks: Can the digital objects, identified in chapter 2, be modelled in a generic way; that is, can it be independent from the specific implementations in software tools and panel studies? If possible, generic solutions are proposed.

A special class of digital objects are metadata, which are usually part of the documentation of a panel study and are typically collected after the corresponding digital objects (like questionnaires or datasets) are created. In metadata communities like the Data Documentation Initiative (DDI), the idea of reusing metadata in subsequent iterations of the research lifecycle (or waves in the context of a panel study) led to a new understanding of metadata, assuming that it might even be possible to automate significant parts of data production in a *metadata-driven* design. Chapter 4 outlines a metadata-driven infrastructure based on the GLBPM and discusses whether it has the potential to actually increase the quality and the usability through automation, asking: How can the concept of a metadata-driven infrastructure optimize the production of panel data and increase the quality and usability of the resulting data?

One significant input to metadata-driven data processing are details about data transformations and related variables. Another use case for the documentation of related variables is the combination of multiple data sources based on statistical matching or record linkage (the combination of multiple waves from one panel study is considered a special case of record linkage). Chapter 5 takes a closer look at use cases for variable linkage and presents solutions implemented by panel studies or the DDI standard, asking: Which information about related variables do survey researchers require to analyse distributed data sources (e. g., statistical matching or record linkage) and how can they be covered in the metadata? The discussion recommends two complementing solutions: concepts for omnidirectional relations and process documentation for unidirectional data transformations.

After the mostly theoretical discussion in part I, part II implements significant parts of the theoretical discussion. In the context of this dissertation, the implementation is, first of all, intended as a proof of concept for the theoretical results. Furthermore, significant parts of the implementation—in particular, the data portal DDI on Rails—are actually used in production to document panel studies. The proof of concept is therefore not only a technical implementation but includes organisational examples.

Chapter 6 designs a reference architecture for a metadata-driven infrastructure using existing open source products. The discussion refers to two related projects: the implementation of a test case including a set of fictitious (non-sensitive) panel data, and the development of the R package r2ddi to extract metadata from research datasets. Furthermore, the reference architecture and the test case provide a

test environment for the development of DDI on Rails, described in the following chapter.

The main application is the data portal DDI on Rails, which is designed to support researchers analysing panel data. Chapter 7 describes the design of the system and how the work from part I influences the implementation. DDI on Rails is already in production on paneldata.org [41], documenting the SOEP Core study, SOEP-related studies, and external studies—which allows a first assessment of the software and the underlying concepts in regard to existing use cases.

Part I

Framework

Chapter 1

Users and producers of panel data

There is increasing interest in why researchers share their data as well as how to facilitate and encourage data sharing [9, 34, 42]. Most work focuses on individual researchers sharing data and not on institutionalized data producers, like long-running panel studies. The context for this dissertation is, however, panel studies, which are usually designed as long-term projects with a considerable amount of funding in order to ensure that representative samples are drawn and preserved over time. In the field of panel studies, methodological researchers concentrate on panel characteristics like attrition, intervention or interviewer effects, and the analytical designs. Hardly any research looks at the management of panel data, what the requirements of the data users are, or at the documentation of specific aspects of panel studies.

We start with a discussion of panel studies from the perspective of data sharing, taking into account two perspectives: the perspective of the data producing institution and the perspective of the data using researcher. While the following chapters take a generic perspective at panel studies, this first chapter is based on the German Socio-Economic Panel (SOEP) [43] as a specific example of a household panel. The research question is: Who are the users and producers of the SOEP data and what are their requirements regarding usability, data quality, and software support?

The chapter starts with a generic framework for data sharing to create a frame of reference for further discussions. The following two sections take a closer look

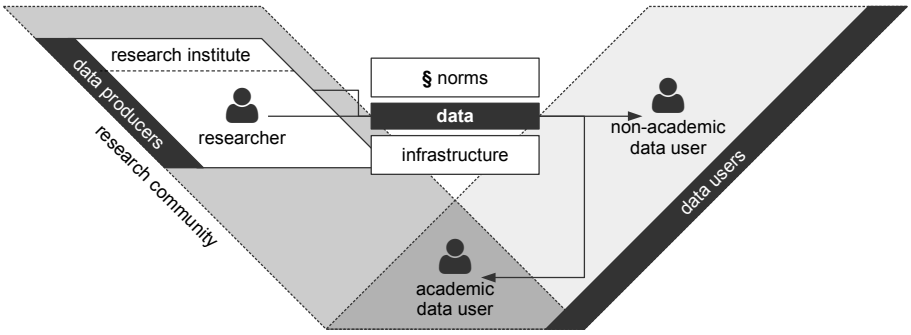

Figure 1.1: Framework for academic data sharing (modified version, based on Fecher et al. [34]).

at the data producing infrastructure and the SOEP users. Due to its data protection policy, re-distribution of the SOEP data is prohibited. The next section discusses the options to enable reproduction and replication of analysis based on the SOEP data. The chapter closes with a discussion of data quality and usability (including documentation, accessibility, and interoperability).

1.1 Framework for academic data sharing

In "What drives academic data sharing?" [34] we[1] identify critical factors influencing an individual researcher's willingness to share data and propose a framework for further discussion (see figure 1.1). The paper is based on the systematic review of 98 scholarly papers and the quantitative analysis of a question module that was integrated in the 2013 SOEP user survey (see section 1.5 for more details on the SOEP user survey). The systematic review covers scholarly articles addressing data sharing in academia. It was used to design a framework consisting of six descriptive categories, including 17 sub-categories. After designing the framework, we test it with the results from the SOEP user survey, including one closed question (general disposition to share data) and two open questions (barriers and enablers regarding data sharing) on data sharing. The quantitative analysis was able to confirm the original framework.

[1] Joint work with Benedikt Fecher and Sascha Friesike.

The framework considers researchers who produce data as part of their scientific work but most notably aim to publish recognized articles—publishing their data is usually less important to them. In contrast, most panel studies are designed and funded to produce panel data that are reused by external researchers, similar to the rare case of individual researchers who obliged by their research organisation or research community to publish their data. Due to the modified context, the original framework is modified for this dissertation in two aspects. First, the terms 'data donor' and 'data recipient' seem misleading in a context where sharing is mandatory (e.g., large panel studies, which are explicitly funded to produce reusable data). The terms are therefore replaced by 'data producer' and 'data user'. Second, the term 'data producer' now includes not only individual researchers but also research institutions, covering use cases like the SOEP.

The data producer can be either an individual researcher or a research organisation. If the data are provided by an individual researcher, his or her probability to share data are influenced by social-demographic factors, the degree of control over the data and their usage he or she retains, the resources required to publish the data, and the returns for sharing them. Both the researcher's organisation and funding agencies can influence the individual decision to share data. In the context of this dissertation, however, the data producer can also be a research organisation that does not depend on the individual researcher's preferences but on the institution's goals and funding terms whether to share data. While the original paper focuses on individual researchers, panel data are usually collected and managed by research institutions. Thus, the following discussion assumes that the data producer is not an individual researcher but a research organisation.

Three groups of factors frame the provision of data: norms, infrastructure, and community. Ethical and legal norms concern both the content of the data as well as ownership. The infrastructure includes the technical and organisational architecture to provide data, including factors like performance, security, support, or accessibility. The research community defines the overall data sharing culture (including community-specific norms), standard for both data and metadata, the scientific value associated with shared data, and demands of journals and publishers regarding the publication of data.

The data recipient or data user can be discussed from two perspectives: either from the perspective of the data *producer* who anticipates a certain user behaviour or from the perspective of the data *user*, looking at his or her expectations and needs. On the producer side, individual researchers are particularly afraid of neg-

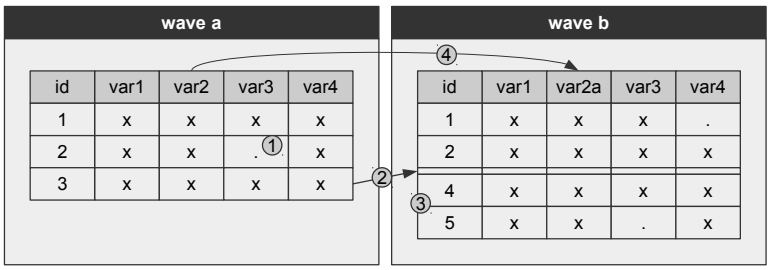

Figure 1.2: Cross-sectional design of panel data, illustrating common problems which are related to the survey design: (1) missing values (indicated with an dot), (2) panel attrition (participants leave the panel), (3) refreshment samples (new samples are added in subsequent waves), and (4) modification of measures over time (e. g., adding a response option to a question).

ative consequences. These concerns usually involve either the risk of adverse use of the data by the recipient or concerns about the recipient's organisation (like insufficient security standards). The opposed perspective of the data user was not covered by the original framework. To fill this gap, section 1.5 takes a closer look at the SOEP data users, analysing the SOEP user survey. This includes, in particular, user requirements regarding shared data.

1.2 Panel data

Three key features distinguish panel studies from other survey designs [25, 44]: First, panel studies are based on repeated data collections. Second, the samples are supposed to be stable over time. And third, the instruments are producing comparable measures over time. Figure 1.2 illustrates how these three aspects shape the resulting data and highlights specific problems like panel attrition or changing measures.

Data collections are usually conducted in constant intervals during distinct fieldwork periods referred to as *waves*. Typical frequencies for waves vary between weeks and years. Although the panel design does not depend on the concept of distinct waves (like the example of the German Longitudinal Election Study (GLES) [45] and its *rolling panel design* illustrates), for the rest of the dissertation, we assume the case of distinct waves with constant intervals.

Unlike other longitudinal designs, panel studies are based on stable samples—the same individuals or entities are surveyed repeatedly. In reality, however, samples constantly shrink due to various reasons, including changes in the social reality (e. g., people leaving the survey context or dying) or methodological problems (e. g., participants start refusing to participate). This process, called *panel attrition*, in combination with optional refreshment samples creates a specific challenge for the analysis of panel data and accordingly for the documentation [46].

Social reality not only affects the sample but also forces survey methodologists to adapt their instruments to external changes. Concepts like 'unemployment' are subject to political reforms and other developments that have to be taken into consideration. Because the analytical power of panel studies depends on repeated measurements that are comparable, panel providers are, at least, responsible to document those changes in a way that researchers can harmonize variables *ex post*.

The conceptual design of a panel study is a specialisation of a longitudinal study (repeated measure but not necessarily repeated samples), which itself is a specialisation of a generic survey (not necessarily repeated data collections). Further, cross-sectional surveys are never conducted in complete isolation, but refer to previous work, with subsequent research projects possibly referring back to them. Therefore, cross-sectional surveys have an iterative moment as well. This will allow us to generalize most of the findings in this dissertation for longitudinal or even cross-sectional designs, even if the discussion focuses on panel studies.

1.3 German Socio-Economic Panel (SOEP)

The original proposal for the Socio-Economic Panel was made in 1982 [47]. It argued that Germany, at that time, was confronted with significant socio-demographic challenges that could not be analysed adequately with existing data sources but required panel data to model developments. It proposed to start with a sample of 5,000 households in Germany, covering a variety of topics, including income, employment, education, living conditions, health, and life satisfaction. In fact, the SOEP started in 1984 with a sample of 5,921 households; in 2012, the SOEP reached 12,322 households [48].

The household interviews are complemented with individual interviews of all adults (age 16 and above) and additional interviews depending on the interviewing situation and the household composition (e.g., special questionnaires for new households or children under the age of 16.). The data collection including the im-

plementation of the questionnaires is conducted by the fieldwork organisation TNS Infratest of Munich. As of 2015, questionnaires come in three versions: on paper, on interview laptops, and as web surveys. The SOEP interviews are usually personal interviews where the interviewer visits the respondents at home and conducts the interview using either a printed questionnaire or an interview laptop. Web-based interviews are being tested as a third option.

The SOEP uses a centralized database (SIR [49]) to store its data and export them into various data formats including the formats of common statistical packages (e. g., Stata or SPSS)—the design is similar to a data warehouse system [50]. As one part of the data service, researchers at the SOEP generate additional variables to make the published datasets more user friendly. The resulting data are published in a major data 'distribution' once a year, supplemented with additional bug fixes as necessary. Each distribution contains all data since 1984, except for a small set of variables (such as spacial data) which are too sensitive to be released—sensitive data are available for on-site usage or remote analysis. The data are documented in the data portal SOEPinfo [51], with additional material available on the SOEP Website [52].

The SOEP data are analysed by both SOEP employees and external researchers. All data users are obligated to report resulting publications to the SOEP where they are managed and published in SOEPlit [53, 54]. The SOEP is evaluated regarding its scientific output (assessed based on the scientific publications) but also regarding its service. As a part of the Leibniz association, the SOEP is evaluated on a regular basis.

The SOEP is located at the German Institute for Economic Research (DIW Berlin) and is divided into three sub-departments: survey methodology, applied panel analysis and knowledge transfer, and data operations and research data center [55]. The three departments illustrate that the SOEP is not only a producer of panel data but also a research infrastructure for panel data and a research institute that is expected to produce scientific results in the form of papers. Usually, researchers at the SOEP are expected to spend half of their time on service (e. g., the generation of user friendly variables) and the other half on scientific work (writing papers).

The fact that researchers provide large parts of the SOEP service (such as data management and data generation) has implications for the research infrastructure. In particular, they are usually trained as social or economic researchers, familiar with rectangular datasets as the common format for research data. Hardly any of these researchers has experience with programming languages (like Java

or Python), non-rectangular data formats (like XML), or database management (like SQL). The training of researchers is a significant constraint for the design of a metadata-driven infrastructure like in chapter 4. The development of DDI on Rails (chapter 7) considers this constraint in mapping significant parts of the XML-based DDI standard to rectangular formats—a format that is familiar to those researchers and therefore gains a higher acceptance.

1.4 SOEPinfo, DDI on Rails, and paneldata.org

Since the late 1990s, SOEPinfo is a central part of the SOEP service [56]. SOEPinfo provides a web-based documentation of the SOEP data on the level of variables and questions. Based on the item correspondence list [23, 57], SOEPinfo supports researchers to find related measures over time. The system includes a variable basket, in which the researchers can collect variables. Afterwards, the script generator provides the corresponding script (Stata, SPSS, or SAS code) to select and merge the SOEP data according to the selection in the basket. With more then 300 datasets available for SOEP-Core, the script generator is a convenient service for the data users.

The development of DDI on Rails was initiated because SOEPinfo was not capable to document studies other then SOEP-Core [58], not even SOEP-related studies like SOEP-IS. DDI on Rails is a part of this dissertation and serves as a proof of concept in the second part. The primary goal is to replicate the functionality of SOEPinfo but in a study-independent manner. The use of the DDI standard (introduced in section 2.1) facilitates a generic data model and, therefore, a re-usable design of the application. The functionality of DDI on Rails, however, exceeds the original SOEPinfo in various aspects, including support for statistical matching and record linkage (see chapter 5), the documentation of multiple versions of one study, and a sophisticated search index.

DDI on Rails is used in production on paneldata.org [41], documenting the SOEP Core study, SOEP-related studies, and external studies. For this dissertation, it is important to distinguish three names: (1) “SOEPinfo” refers to the original system, developed in the late 1990s; (2) “DDI on Rails” is the new software and the software only, which is designed as a successor for the original SOEPinfo; (3) DDI on Rail is used in production on “paneldata.org”, where data users can find the current documentation of the SOEP and other studies. DDI on Rails and paneldata.org are discussed in more detail in chapter 7.

1.5 Secondary data users

Since 2011, the SOEP user surveys are conducted annually in order to gain insights about researchers working with panel data. Earlier user surveys were irregular, with only the 2004 user survey providing quantitative data as a reference. We[2] provide a general overview of the results in "On the Structure of Empirical Social, Behavior, and Economic Researchers Using the SOEP: An Overview of Results from the SOEP User Survey" [38].

The web-based survey is implemented in LimeSurvey [59] targeting all SOEP users. The following analysis focus on the 2013 survey but includes results from the other years as well. The 2013 survey resulted in 603 valid interviews. Contrasting this number with approximately 1,250 active data users (researchers who received the SOEP data or a data update in the last five years), we estimate an response rate of almost 50 percent. The questionnaire consists of demographic questions, general question about the SOEP data and service, as well as varying questions on specific topics like the documentation infrastructure or the usage of the longitudinal aspect of the data. The participants are on average 37 years old and comprise of more men then women (60 % men). The most represented disciplines are economics (43 %), sociology (36 %), and psychology (6.5 %). Sixty percent of those surveyed have experience in academic teaching.

The following analysis complements our paper [38] and takes a closer look at three question concerning the following chapters of this dissertation: (1) How do the SOEP users analyse the research data? (2) Which software is used for the analysis of the data? (3) What functionality do the users expect in a metadata portal like SOEPinfo?

Regarding data analysis, it is worth mentioning that the SOEP samples are drawn on the household level, but all (adult) members of the households are also interviewed individually. Despite the fact that the SOEP is a household panel, most researchers focus on the individual level (97 %), with only 57 % explicitly analysing at the household level. The majority takes advantage of the specific characteristic of a panel study and analyse the data over time (84 %). However, 77 % generate cross-sectional statistics as well. Regional or spacial data are used by 34 %, even if those data have a more complex structure and are limited in their accessibility. Regarding the longitudinal design, the wide format was considered the preferred data structure for a long time, but the majority of the 2013 respondents solely uses

[2] Joint work with Florian Griese, Janine Napieraj, Marius Pahl, Carolin Stolpe, and Gert Wagner.

the long-format (55 %), 25 % use both the long and the wide format, and only a minority of 12 % uses solely the wide format. In 2010 the SOEP released a beta version of harmonized data in the long format, which was used by 35% of the users responding in the 2013 survey.

Most researchers work with Stata (76 %) and SPSS (30 %) to analyse the data. The comparison of the data from 2004 to 2011 illustrates how fast these preferences can change: In 2004, SPSS was the most popular package, changing position with Stata in less then 10 years. Stata on the other hand, the dominant package in 2011, had not yet been released when the SOEP started in 1984. (Stata version 1.0 was released in 1985 [60].) SAS, once one of the most popular packages, is now down to only 3.5 %. The statistical package currently gaining users the fastest, is the open source software R (no users in 2004, 15 % in 2013). The software support for SAS in SOEPinfo was therefore replaced with support for R in DDI on Rails when it was released to the public in December 2013. The fluctuation indicates the fact that these statistical packages and their proprietary formats are too short-lived to be used for long-term preservation.

SOEPinfo supports researchers searching for variables, provides additional information about those variables (such as links to related variables, frequencies, or the underlying questions), and allows the user to collect variables in a basket to export them in various statistical scripting languages (Stata, SPSS, and SAS). The 2011 user survey asked researchers how useful these functions are. Table 1.1 provides an overview, sorted by the researchers' preferences. The most important functionality is the variable search (92 %), followed by the internal links to questions (85 %) and corresponding variables over time (83 %). It comes as a surprise that 42 % of the users do not think that the basket and the related syntax generator provide a helpful functionality.

In summary, the SOEP data users are mostly social, economic, and behavioural researchers that normally analyse panel data in the long format and on the individual level. Based on the 2013 survey, they prefer Stata, SPSS, and R to analyse the data; but the comparison with the 2004 user survey makes clear that preferences change from time to time. The following chapters, in particular when designing the reference architecture in chapter 6 and introducing DDI on Rails in chapter 7, will respect the reported 2013 preferences but also consider the fact that preferences are likely to change.

Table 1.1: Assessment of the functionality in SOEPinfo. The original question used a five-point Likert scale that was recoded for the reader's convenience: helpful combines very helpful and rather helpful, neither remains, and not helpful combines rather not helpful and not helpful at all. The categories are ordered by the researchers' preferences.

		helpful	**neither / nor**	**not helpful**
1	variable search	91.72	7.10	1.18
2	questions	84.83	11.38	3.79
3	corresponding variables	83.45	14.03	2.52
4	frequencies	77.04	16.04	6.92
5	topics	63.31	25.54	11.15
6	basket	62.50	24.29	13.21
7	dataset composition	58.30	26.20	15.50
8	syntax generator	58.14	24.03	17.83

1.6 Data sharing and re-analysis

John Chambers [61] describes how users of statistical software packages often transition, becoming programmers in order to express their ideas computationally. This section assumes that a similar transition from secondary data user to data producer is common in social and economic sciences. Scientific results are expected to be reproducible or even fully replicable [62, 63]. Thus, the first reason for sharing data is to enable researchers to verify results in re-analysis. Further, examining the data might reveal for new research questions. Sharing data supports other researchers and makes scientific progress more efficient. Nevertheless, researchers sharing data must consider various limitations, in particular privacy issues.

Panel studies like the SOEP limit the access to their data to only those researchers who agree to abide by data protection policies that protect the privacy of participants. Therefore, researchers are usually not allowed to publish the data after processing them for analysis, even as journals and researchers increasingly demand publication of underlying data [64]. The first step to enable the reproducibility of analyses, however, is not the data but rather the processing and analysis scripts, which are less sensitive and can usually be published without any limitations other than copyright. Persistent identifiers (see section 5.1) allow researchers to reference data sources. The combination of the data (or persistent identifiers for the data)

and the corresponding analysis scripts is considered to be valid documentation of an analysis project, which enables reproducibility [62].

Data repositories support researchers with both the technical infrastructure to share data and the organisational infrastructure necessary for restricted access data. The later includes, for example, on-site access to sensitive data or the necessary contract management for data users, which are two things that are nearly impossible for an individual researcher to provide. As an example, the SOEP provides an archive as a service within its research data center (RDC) [65]. In this archive, researchers can store material related to their analysis of the SOEP data.

1.7 Data quality and usability

Data *quality* refers to the content of the data—whether they adequately represent a particular aspect of reality that they refer to. The data quality depends on the initial study design, the samples, the instruments and methods, the data transformations that applied, as well as other aspects of survey methodology and data management. Lynn [66] proposes an comprehensive framework to assess the quality of longitudinal data. Nevertheless, data quality is a necessary but insufficient condition for a researcher to draw correct conclusions. The researcher must also find the appropriate data and variables as well as be able to understand and interpret their content correctly. Data *usability* refers to the fact that a data user is able to utilize the data. It depends, most of all, on suitable documentation of the study and the data.

Sections 1.3 and 1.5 discussed the users and producers of the SOEP data. Both groups mostly consist of social and economic researchers with a profound knowledge of their preferred statistical packages but, at best, rudimentary knowledge of programming languages and complex data structures. They are used to rectangular datasets as the preferred data structure for statistical analysis. The experience and knowledge of the researcher is a crucial restriction for the development of a metadata-driven infrastructure (chapter 4) and the development of the reference architecture (chapter 6).

Chapter 2

Generic process model

Chapter 1 provides an introduction to the requirements of researchers using and producing the SOEP data. The SOEP, however, is only one panel study amongst many others, both conducted by the SOEP organisation and by entities around the world. The SOEP conducts additional surveys designed to be interoperable with the core study. The surveys include an additional panel for innovative methods and other studies that are funded by third-parties. Panels are also conducted by other organisations, both within Germany (e. g., Pairfam, GIP, and NEPS) and around the world (e. g., PSID, Understanding Society, and Hilda). The following chapters and the development of DDI on Rails are intended to be reusable in the context of panel studies other than the SOEP.

To bring the discussion on a more abstract level, this chapter develops a generic model of panel studies, asking: What does a generic process model for panel studies look like and which digital objects can be identified in the model? The results should not only support the development of DDI on Rails but also the development of re-usable generic tools and the design of interoperable workflows for panel studies.

The core of this chapter is the Generic Longitudinal Business Process Model (GLBPM). The first two sections describe the context for the development and use of the GLBPM. First, the history of the Data Documentation Initiative provides essential background information on the DDI standard as one of the most important inputs to this dissertation and, in particular, highlights the emergence and relevance of generic process models for the effective design of research infrastructures. Second, we define a set of reference studies to tested the model and to ensure the

re-usability of further results. The Generic Longitudinal Business Process Model (GLBPM) covers three sections: the introduction of the model from a historic perspective, the presentation of the major nine phases in the model, and the identification of digital objects in the model. The chapter closes with a discussion on how to utilize a generic model like the GLBPM, and the implications of generic models for a sustainable design of panel studies and supporting tools.

2.1 Background: Data Documentation Initiative

The Data Documentation Initiative (DDI) [67] provides a widely used metadata standard for the social, economic, and behavioural sciences. Section 4.5 takes a closer look at the design of the standard itself. For now, we focus on the history of the initiative and the development of the standard through earlier versions [1, 68]. This provides not only the context for the introduction of the standard, it also introduces the use and value of generic process models from a historical perspective.

In the 1960s and 1970s—long before tools like Stata, SPSS, or SAS integrated variable and value labels into their file formats—, datasets (e. g., in the OSIRIS format) usually consisted of numeric values representing categorical measures and the meaning of the numeric representations was stored separately in complementary codebooks. Blank [21] points out that the technical infrastructure was very homogeneous at that time (in particular, OSIRIS running on IBM mainframes) and metadata were stored in an almost standardized manner. This changed fundamentally during the 1980s when the PC became powerful enough for data management and analysis. With diverging platforms, operating systems, and statistical packages, multiple designs for data and metadata management emerged. (Section 3.3 provides a detailed discussion of problems that result from conflicting data models in various statistical packages.)

In the 1990s, the potential of and the need for standardized data to facilitate interoperability became more obvious as the Internet was increasingly used to exchange data. In 1993, a group inside the International Association for Social Science Information Service and Technology (IASSIST) initiated the development of an XML standard for codebooks. In 1995 this work transitioned into the formation of the Data Documentation Initiative (DDI), which published version 1 of the DDI standard in 2000 [69]. This version broadened the perspective on metadata and documentation. This first and the following second version are now referred to as the 'codebook' branch of the standard. However, both versions include context in-

formation about the study design, the authors, the instruments, and other aspects of data production.

After publishing version 2 in 2003, the initiative started to fundamentally rethinking the principles and the design of the standard: The DDI Lifecycle model (see figure 2.1 on page 29, introduced in section 2.3) became the foundation for the design of the standard [1]. The most important change might be that metadata curation is no longer an *ex post* task for data production. It is integrated into the whole process and metadata can be reused in the iterative design of the lifecycle. In 2011, work on the next version of the standard started under the working-title 'moving forward' [33]. In particular, the previous versions are pure XML standards, neglecting the importance of relational databases, object orientated software design, and other data formats. The next version will be *model-based*—the standard will be designed in the form of UML diagrams (models) first, complemented by written documents, and finally resulting in various technical implementations (including XML Schema Definitions).

For this dissertation, the history of the DDI standard highlights the need for standardized metadata and with careful study revealing two important observations about the DDI standard: First, the transformation from a static codebook design to the dynamic lifecycle design has gone far but is not yet fully implemented. Second, the current XML version of the standard is not lossless transferable to relational databases and object orientated design, which will affect the design of DDI on Rails in part II.

2.2 Reference studies

Before designing a generic reference model, table 2.1 provides a set of reference studies that should fit the generic model. The studies are grouped in three categories: SOEP-related studies, other panel studies in Germany and Europe, and household panels included in the Cross National Equivalent Files (CNEF). The work in this dissertation and the development of DDI on Rails is intended to be re-usable for at least this set of studies.

SOEP-related studies are designed to be interoperable with the core SOEP, five are part of the reference set. The SOEP Innovation Sample (SOEP-IS) started in 2011 by extracting two samples from the core study [70] and includes more innovative research instruments. SOEP-IS is a panel study, considered to be an expansion of the former SOEP Pretests, which were designed as cross-sectional studies. Families

Table 2.1: Studies related or similar to the SOEP: (1) SOEP-related studies (located or co-located at the DIW Berlin), (2) other panel studies (in Germany and Europe), and (3) the household panel surveys included in the Cross National Equivalent File (CNEF).

	Study
1	SOEP Innovation Sample (SOEP-IS) [70] and SOEP Pretests
	Berlin Aging Study II (BASE-II) [71, 72]
	Families in Germany (Familien in Deutschland, FID) [73, 74]
	PIAAC Panel (PIAAC-L) [75]
	TwinLife [76]
2	National Educational Panel Study (NEPS) [77]
	Pairfam [78]
	Survey of Health, Ageing and Retirement in Europe (SHARE) [79]
	German Internet Panel (GIP) [80]
	Processes of Mate Choice in Online-Dating (PPOK) [7]
3	Understanding Society (UK, former BHPS) [81]
	Household, Income and Labour Dynamics in Australia Survey (HILDA) [82]
	Korean Labor Income Panel Study (KLIPS) [83]
	Panel Study of Income Dynamics (PSID, USA) [84]
	Russia Longitudinal Monitoring Survey (RLMS-HSE) [85]
	Swiss Household Panel (SHP) [86]
	Survey of Labor and Income Dynamics (SLID, Canada) [87]

in Germany (FiD) [73, 74] was initiated by the Federal Ministry for Family Affairs, Senior Citizens, Women and Youth (BMFSFJ) and the Federal Ministry of Finance (BMF) to evaluate the full range of public benefits for married people and families. After the funding expired, both the FiD sample and the existing data are integrated into the SOEP-Core study. This provides a use case for linking and merging panel data, discussed in chapter 5.

Further SOEP-related studies are TwinLife [76], PIAAC-L [75], and the Berlin Aging Study II (BASE-II) [71, 72]. They are externally funded and combine a panel survey with other designs for data collection. TwinLife is a behavioural and genetic study investigating the development of social inequality based on a sample of 4,000 twins. PIAAC (Programme for the International Assessment of Adult Competencies) is a worldwide OECD survey on adult skills. PIAAC-L or PIAAC Panel continues a sample of more than 5,000 respondents from the German PIAAC survey of 2011/12. This project is a cooperation of Gesis, the National Educational Panel Study (NEPS), and the SOEP. Besides the SOEP questionnaires, competency measures from both PIAAC and NEPS are used. BASE-II collects data on the objective health (e. g., cardiovascular system, musculosceletal system, immune system), the functional capacity (e. g., physical capacity, vision, hearing, balance), and the subjective health and well-being of the respondents.

The panel design is popular in Germany. Two panel studies, the family panel Pairfam [78] and the German Internet Panel (GIP) [80], intend to use DDI on Rails for their data documentation. Thus, they are the most relevant external use cases here.

The 2008-launched family panel Pairfam (Panel Analysis of Intimate Relationships and Family Dynamics) [78, 88] is a multi-disciplinary, longitudinal study on partnership and family dynamics in Germany. The annually collected survey data from a nationwide random sample of more than 12,000 persons of the three birth cohorts 1971–73, 1981–83, 1991–93 and their partners, parents and children, offers unique opportunities for the analysis of partner and generational relationships as they develop over the course of multiple life phases.

Designed as an infrastructure project, the German Internet Panel (GIP) [80] collects data about individual attitudes and preferences that are relevant for political and economic decision-making processes. The data form the empirical basis for the scientific research of multiple SFB 884 (Political Economy of Reforms) project groups. The methodological composition of the GIP aims to build a panel study that, on the one hand, benefits from the advantages of online surveys (lower costs,

higher flexibility) and, on the other, is representative for the entire German population (age 16 to 75). Currently about 1,500 respondents participate in the study and are invited bimonthly to participate in a survey. Topics include family, policy and economy.

The National Educational Panel Study (NEPS) [77] samples specific age cohorts (starting with *Kindergarten* through adulthood). The Study on Health, Ageing and Retirement in Europe (SHARE) [79] complements traditional survey design with the collection of biomarkers. Moreover, the fieldwork across Europe involves multiple organisations, increasing the organisational complexity.

In comparison to all previous studies, the project Processes of Mate Choice in Online-Dating (PPOK, Prozesse der Partnerwahl auf Online-Kontaktbörsen) [7] was significantly smaller and has already expired. Nevertheless, the PPOK project is interesting for two reasons. First, it provides a reference to discuss whether the generic framework in section 2.3 works for small projects. Second, the project was originally not designed as a panel study but used process generated online data. The panel was implemented later to complement these data, providing a use case for variable linkage in chapter 5.

The *Cross National Equivalent File* (CNEF) [89, 90, 91] consists of eight household panels from the United States, Germany, Great Britain, Canada, Russia, Korea, Swiss, and Australia. Unfortunately, the Canadian household panel has been expired. CNEF data provide a basic set of variables harmonized for the included studies. The group of studies in the CNEF represent an important trend in survey design, the concept of household panels. These surveys sample on household level, but also interview individual members (usually with some age restrictions) covering a broad selection of topics primarily from the social, behavioural, and economic sciences but also from demography, geography, and psychology.

For further discussions on how to implement re-usable tools and to design metadata-driven infrastructures for panel studies, it would be overwhelming to take all specifics of these reference studies into account. To use, in contrast, the SOEP as the sole use case would significantly restrict the possibility to re-use and generalize the results. The proposed solution abstracts from the various implementations and designs a generic model of panel studies, to build the following discussion on it.

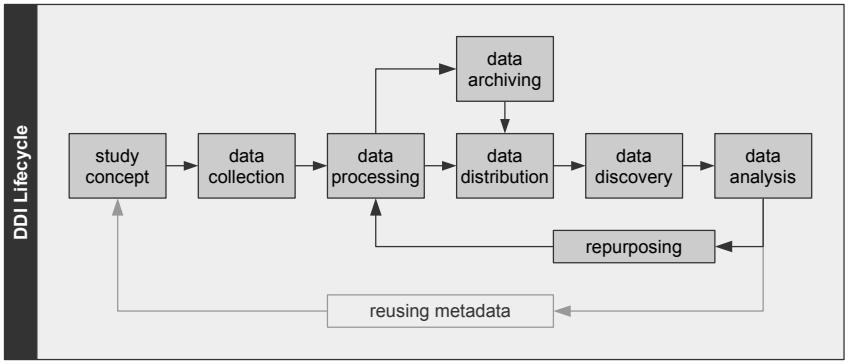

Figure 2.1: The DDI Lifecycle model [1], modified to highlight the iterative character [94].

2.3 Generic Longitudinal Business Process Model

The DDI Lifecycle (figure 2.1) provides an abstract model of the data flow in a research project. The model is simple but still sufficient to improve the development and quality of the DDI standard (version 3) significantly. The DDI Lifecycle is not considered to be a linear process starting with the study concept and ending with the data analysis but rather an iterative design. Metadata created in one iteration can be reused in subsequent iterations, reducing the overall costs of documentation. The 1st Dagstuhl Workshop on Longitudinal Data started refining this and other longitudinal aspects of the standard [92, 93, 94, 95].

Based on the DDI Lifecyle model, a group of national statistical institutes (NSIs) under the direction of the United Nations Economic Commission for Europe (UNECE) developed a more sophisticated business process model for the collection and generation of statistical data, called the Generic Statistical Business Process Model (GSBPM) [96, 97]. Unfortunately, the workflows in NSIs and in academic panel studies differ significantly. Thus, at the 2nd Dagstuhl Workshop on Longitudinal Data, we[1] designed a modified version of the GSBPM, optimized for longitudinal and panel studies. The result is the Generic Longitudinal Business Process Model (GLBPM) [37].

[1]Ingo Barkow, William Block, Jay Greenfield, Arofan Gregory, Marcel Hebing, Larry Hoyle, Wolfgang Zenk-Möltgen

Like its statistical counterpart (the GSBPM), the GLBPM consists of nine major phases, accompanied by continuous tasks (see figure 2.2). On this level, the biggest adjustment is the integration of data analysis (referring to analysis that test the quality of data) into the data processing phase and adding an additional phase, *research and publish*, thus highlighting the most important difference between NSIs and academia: the way scientific work is created and rewarded. The nine top-level process steps represent the generic design of panel studies. The following discussion is based on the top-level steps and, only if required, more detailed steps will be introduced. The original paper [37], however, provides a detailed introduction of all sub-steps.

2.4 Phases and process steps

The original GLBPM work is vague on the objects that are managed throughout the whole process. The identification of the nine major process steps is supplemented by the introduction of digital objects that are required, processed, or created in specific steps. The term 'digital objects' includes all kinds of objects including web-based questionnaires, images, datasets, and scientific articles.

In *phase 1*, the needs of the project are evaluated and specified. Most of the material in this phase are written documents containing research questions, project descriptions, or funding applications. This is the phase with the least amount of structured metadata. The first wave can only refer to external material (see continuous task *use of external standards*). Indeed, every study depends to some degree on previous and external work. These dependencies can be documented as citations to external work. In subsequent waves, the results from phase 9 (retrospective evaluation) might become the most relevant input. The results of the evaluation and the specification of needs become an important input for the more detailed study design in phase 2.

In *phase 2*, potential data sources are evaluated and the data collection is designed (in subsequent waves: re-designed). In a codebook-orientated documentation environment, this phase is usually not documented in a structured way. The lifecycle approach allows structured metadata, but it will be a major point for discussion whether it is reasonable to document the early phases in a strictly standardized manner. Base on the results from phase 1, the methodology (sampling methods, instrument design, etc.) and the organisational infrastructure are specified. Phase 2, however, operates on a conceptual level and does not implement any

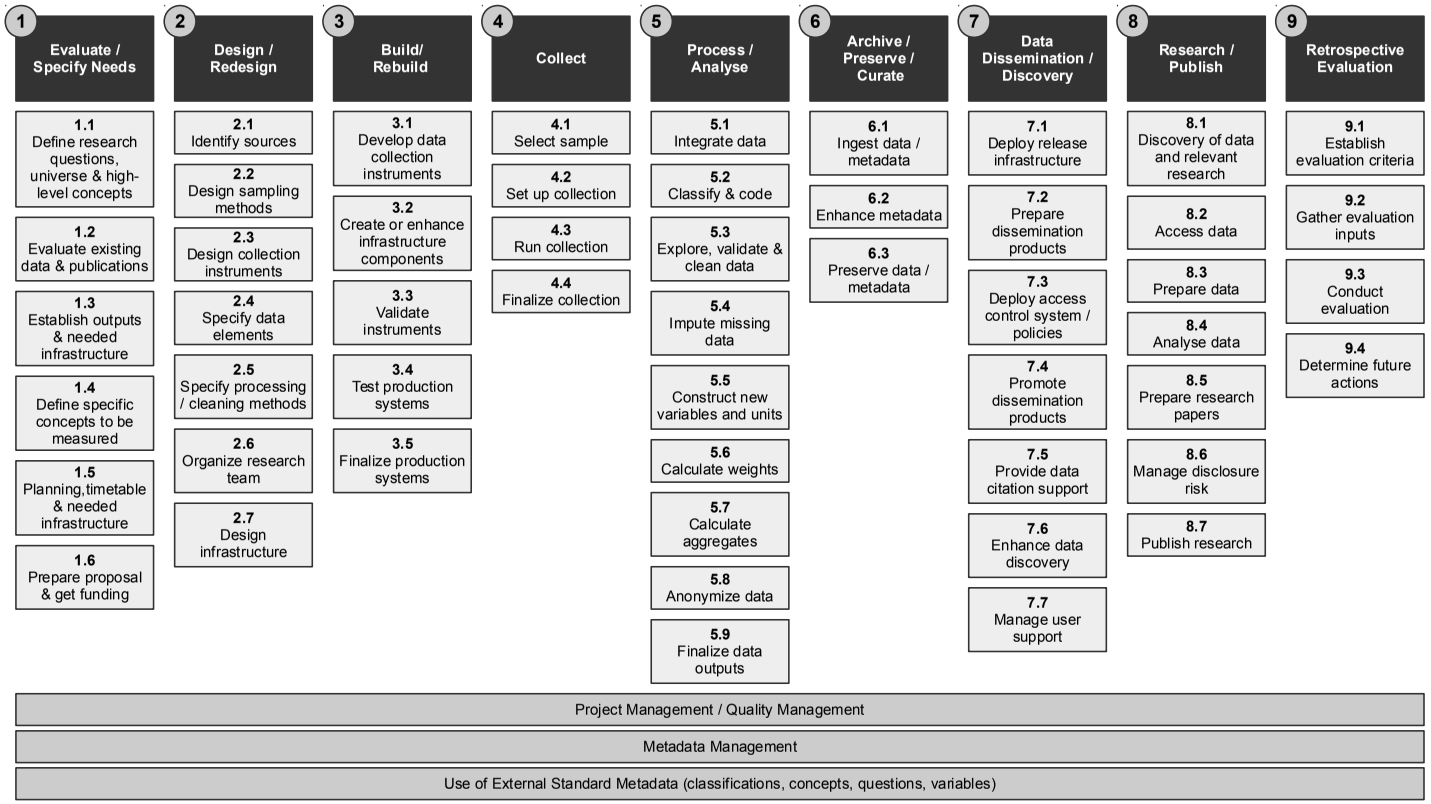

Figure 2.2: The Generic Longitudinal Business Process Model: overview [37].

tools or instruments. The first two phases design a conceptual study description to be assessed in phase 9.

In *phase 3*, the instruments for the data collection are built and later re-built, including validations and tests of the tools. The documentation of the instrument is crucial for the interpretation of the resulting data. The instruments (as the main output of phase 3) are implemented based on the specifications from phase 2. In panel studies, the instruments are usually questionnaires. However, the list of studies in section 2.2 present several alternative data sources and collection designs (e. g., collecting biomarkers or extracting data from administrative sources).

In *phase 4*, the sample is selected (even so the design of the sample belongs to phase 2) and the data are collected. The output of this phase are the raw data. Thus, in this phase we not only document how the collection works, but we also start documenting data files and processing tasks. Based on the sample design from phase 2, phase 4 selects the actual sample and conducts the data collection, using the tools from phase 3. The detailed documentation of the fieldwork (including sample selection and data collection) enables external researchers to assess the quality of the data.

In *phase 5*, the raw data are processed and analysed. We combined phase 5 (process) and 6 (analyse) of the GSBPM into one phase, because the understanding of *analysis* by panel data users differs from the statistical world's meaning. The analysis of the data in this phase is not focused on publishing results but rather on testing and validating the data. Research-focused analysis belongs to phase 8. The documentation of data processing, data generations, and data transformations are discussed in section 3.4. Phase 5 takes the (raw) data produced in phase 4 and transforms them into a data product that can be shared with other researchers for further analysis. Ideally, these transformations are conducted using transformation scripts. In every case, the comprehensibility of shared data depends on the documentation of transformation tasks that generated them.

In *phase 6*, the data are archived, preserved, and curated. The circle view of the GLBPM (figure 2.3, introduced in detail at the end of the section) separates this phase from the other phases and designates the archive as a hub. Multiple phases interact with the archive to store data, but also to retrieve data in subsequent iterations. The archive depends, in particular, on consistent and persistent identifiers, which are discussed in section 5.1. This phase basically takes every object from all the other phases as input to prepare and store for long-term preservation.

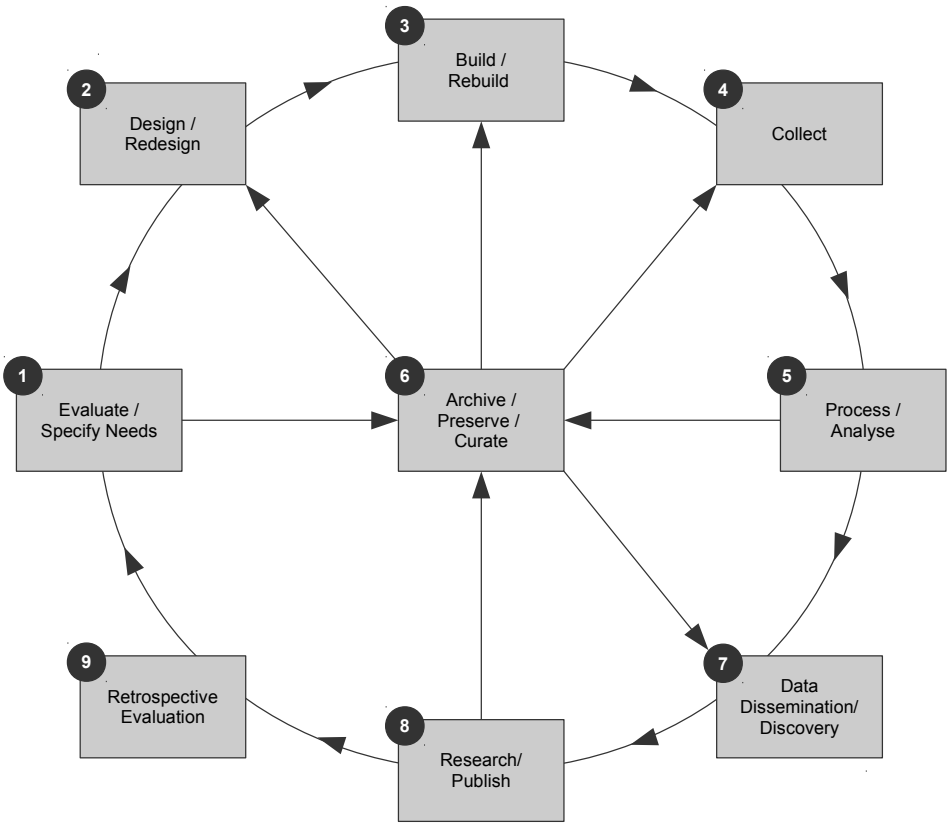

Figure 2.3: The Generic Longitudinal Business Process Model: circle view (modified version, based on [37]).

In *phase 7*, the data are released, with users discovering and obtaining. This might be the phase where high quality metadata pay off the most. The development of DDI on Rails is, most of all, a contribution to this phase. Phases 5 and 6 prepared data and other material for publication and preservation. The actual data dissemination process produces administrative data (information about data users, contracts, etc.).

In *phase 8*, users address their research questions using the data and publish results—in particular, as peer-reviewed articles. The increasing request for reproducible research creates new demands. In addition to the traditional standards for publications, the demand for reproducible research creates new needs. Re-analysis archives help researchers to link their publications with the underlying material, including data and scripts. Beyond the research papers produced in phase 8, this underlying material is also an important output that should be documented, archived, and made public. The documentation of data transformations and analysis scripts is discussed in greater detail in subsequent chapters.

In *phase 9*, the process of a given iteration is evaluated and leads back to phase 1. Metadata and the documentation, generated during the whole process, support the final assessment of an iteration. Based on the conceptual study description from the first two phases, phase 9 evaluates the execution of phases 3–8. This might lead to recommendations for subsequent waves (assessment report) and should result in an updated version of the study description that no longer describes what should happen but what actually happened.

The nine phases are accompanied by three groups of continuous tasks: *Project / quality management* refers to the business side of a panel study. *Metadata management* as a continuous tasks links material from distinct phases together and becomes a prerequisite for the discussion of a *metadata-driven* infrastructure in chapter 4. *Use of external standards* (classifications, concepts, questions, variables) ensures the interoperability of one study with external studies or data sources. Chapter 5 discusses options to link studies on the variable level.

As previously noted, it became obvious during the work designing the GLBPM that the nine steps can be followed in a sequential order except for one: the archiving phase. While all other steps can be modelled using a simple input-output-relation with the previous and the subsequent step, the archive has similar interactions with multiple steps. The horizontal design of the GSBPM and the GLBPM (figure 2.2) hide the iterative character of all these models. Consequently, figure

2.3 provides an alternative design, taking into account both the specific role of the archiving task and the iterative design of the whole model.

2.5 Digital objects and classes

Some of the digital objects identified for the nine phases recur in identical or similar form in other phases. Most notably, multiple phases create or process research data. Table 2.2 lists the digital objects identified in the GLBPM. These digital objects can be grouped in seven classes: study descriptions, instruments, research data, transformation scripts, publications, administrative data, and documentation.

The *study description* groups mostly unstructured documents. This includes details on the study design, methodology and fieldwork, the description of the data and the data access, as well as the final assessment report. Section 3.1 proposes a semi-standardized structure for information included in the study description.

The *questionnaire* is the most important instrument for panel studies and a sustainable documentation of all modes of a questionnaire (e. g., paper and pencil, computer assisted personal interviewing, or web-based questionnaires) can become quite sophisticated. The following chapters focus on questionnaires as a special class of instruments. In particular, section 3.2 discusses the challenges of a generic data model for questionnaires across modes and tools.

Raw data (phase 4), shared data (phase 5), archived data (phase 6) and analysis data (phase 8) represent different stages of *research data* during the data lifecycle. Section 3.3 analyses the characteristics of these stages and chapter 5 presents metadata models to link the datasets across the stages. The transitions from one stage to another is usually automated based on *transformation scripts*, including processing and analysis scripts. Section 3.4 discusses the design and management of transformation scripts.

A significant part of the digital objects and other material is bundled in the *documentation*. Section 3.5 describes the design and content of a substantial documentation. Further sections provide additional details regarding metadata (section 4.3) and identifier systems (section 5.1).

The aim of this dissertation is to design and support a metadata-driven infrastructure for panel studies, resulting in data of good quality and usability. Chapter 3 discusses these first five classes of digital objects in more detail as they are of particular relevance for the overall aim of the dissertation. The remaining objects (including instruments other than questionnaires, non-rectangular formats for

Table 2.2: List of the digital objects identified in the nine phases of the GLBPM (see section 2.4). Each digital object is assigned to a more abstract class.

Phase	**Digital objects**	**Class**
phase 1	external references	study description
	study design	study description
phase 2	methodological report (including sample and instrument design)	study description
phase 3	instruments	instruments
	instrument description	study description
phase 4	raw datasets	research data
	fieldwork documentation	study description
phase 5	processing scripts	transf. scripts
	dataset description	study description
	shared data	research data
phase 6	persistent identifiers and additional metadata	documentation
	archived datasets	research data
	all digital objects	documentation
phase 7	shared datasets	research data
	documentation (including other digital objects)	documentation
	user and contract data	admin. data
phase 8	analysis scripts	transf. scripts
	analysis datasets	research data
	publications	publications
phase 9	assessment report	study description

datasets, publications, and administrative data) are also of significance for science. They are, however, of less relevance for the task at hand and will not be discussed in more detail.

2.6 Utilizing a generic model

The GLBPM provides a generic framework for the design of longitudinal studies, in general, and panel studies, in particular. The design distinguishes nine phases, considers further iterations, and is interoperable with other models like the DDI Lifecycle or the GSBPM. Seven classes of digital objects are identified from the GLBPM, five of which are discussed in more detail in the following chapter. Even without the digital objects, the GLBPM has several use cases, some of which are the development of metadata standards like DDI, the discussion of actual panel studies based on a common vocabulary, and the identification of requirements for software support in panel studies with a focus on reusable development.

The original motivation for the GLBPM was to support the development of the DDI standard. Like its predecessor, the DDI Lifecycle, the GLBPM focuses on the reuse of metadata. The circle view in figure 2.3 highlights the iterative design of the model.

The GLBPM is also useful for the description and design of actual panel studies. The first step in applying the GLBPM is to treat it like a controlled vocabulary. This implies that process designers should reuse the terms suggested in the model instead of creating new ones, which is a common source of confusion. The second step is to describe how process steps are re-ordered in relation to the generic model. This is a very efficient way of highlighting specific characteristics in a particular workflow.

The GLBPM can also be used as a framework for software development. Even if the requirements for a software project should come from real use cases and not from theoretical models, a generic model like the GLBPM can support the development of reusable tools. It basically provides a framework to define and adjust requirements and designs to produce re-usable functionality. Furthermore, it helps software developers to isolate use cases—this focus increases the chance of a development project to deliver useful software on time.

Chapter 3

Digital objects

In the introduction of the Generic Longitudinal Business Process Model (GLBPM) in the previous chapter, five classes of digital objects that require further discussion are identified. For each of these digital objects, there are many software tools available to design, manage, or even execute their content: text editors and specialized tools for study descriptions, statistical packages (like Stata, SPSS, or R) and database systems for research data and transformations, as well as survey tools for questionnaires and other instruments.

Unfortunately, most tools are implemented independently, with individual data models and exchange formats, resulting in inconsistent or even conflicting data structures. Inconsistent data structures interfere with the aim of this dissertation to build a generic framework. Therefore, this chapter looks in more detail at the generic characteristics of each class of digital objects asking: Can the digital objects, identified in chapter 2, be modelled in a generic way; that is, can it be independent from the specific implementations in software tools and panel studies?

In the following, digital objects are discussed on a conceptual level. The conceptual idea of each object is then tested with actual implementations in existing panel studies or established software products. From the conceptual perspective, a research dataset is solely a collection of data points. The actual implementation of a `.sav` file in SPSS, for example, includes additional metadata like multilingual labels or documentation of the questionnaire. However, the more details are integrated into a digital object, the less likely it is that the object will be interoperable among process steps and software tools. To focus on the minimum conceptual idea of a digital object increases the chance of finding a generic and interoperable represen-

tation. The sections in this chapter discuss, individually, the generic characteristics and common implementations of the five most important classes of digital objects from chapter 2: study descriptions, research data, questionnaires, transformation scripts, and documentation.

3.1 Study description

In the GLBPM, the study description bundles information from six distinct phases (see figure 2.2): the study design from phase 1, methodological details from phase 2, the instrument description from phase 3, the fieldwork documentation from phase 4, the documentation of the resulting datasets from phase 5, and the evaluation from phase 9. Furthermore, a study description is expected to given an overview of the whole process of a study (like an abstract). A first set of elements for this purpose is available in the DDI Codebook standard [98], which defines an elaborated set of standardized fields for the description of a study. This includes citation details, the method of data collection, information about data access, and similar information. (For more details, see the field-level documentation of the `<stdyDscr>` element [99] or the tree layout of the DDI-Lite standard [100] for a comprehensive illustration.)

While the set of fields from the DDI Codebook standard provide a reasonable starting point to outline a study description, the highly standardized structure of the standard seems too narrow to represent the variety of information identified in the GLBPM. Therefore, the following proposition for a study description suggests only a high-level outline to be filled with prose text and links to additional material. More structured outlines tend to fail when documenting innovative designs and methods that have not been considered when the structure was defined. The proposed outline for a study description consists of seven elements: abstract, citation, method, data description, data access, study units, and other material / notes.

(1) The study description starts with an overview in form of an *abstract*. Abstracts are an established element of academic publications. Booth et al. [101] suggest three elements for abstracts in publications (context, problem, main claim) that can also be used to describe a study: start with the context of the project, explain the problem that gets addressed, and outline the solution (methods and resulting data).

(2) The second section, *citation*, bundles basic references related to the study. This includes the title of the study, primary investigators, citation standards, and persistent identifiers. The citation follows the claim for "credit where credit is due" [11] and provides the minimum information necessary to cite a study and its data.

(3) The *method* section includes the instruments, the sample design, and the fieldwork documentation. A comprehensive documentation of these elements in one section seems impossible—it is recommended to design this section as a structured list of references to further documents like questionnaires or fieldwork reports. Nevertheless, the reader of this section should get a basic understanding of the methodological design.

(4) The *data description* has two purposes: to document the design choices regarding the structure of the research data (e. g., conventions for missing codes or variable names), and to provide an overview of the available datasets and their content. Codebooks for the individual datasets and data portals like DDI on Rails can complement this section.

(5) The section on *data access* describes rules and options to access the research data. Studies that include detailed information on the individual level (microdata and, in particular, spatial data), often distinguish specific levels of sensitivity and define corresponding access options (e. g., download, remote access, or on-site usage). This section should provide the information to understand what data can be accessed under which conditions.

(6) In a panel study, *study units* usually correspond to individual waves. Some waves have specific characteristics (e. g., new samples or unique instruments) that can be documented in this section. For more complex designs, every study unit can be described in an individual study description, resulting in a nested structure of study descriptions.

(7) The final section, *other material and notes*, is optional. Innovation is an important aspect of academic research and it seems unrealistic that any given outline or structure can anticipate all future designs. This last section provides a place to reference or include material that does not fit in the previous six elements.

The proposed outline implies that the study description results in a combination of text content and additional files. Both can be edited and managed in content management systems (CMS), a class of software products with many reliable systems available. From a technical point of view, the study description is, therefore, the least problematic class of digital objects in this chapter. And many studies already use appropriate systems in production. However, most studies have very

different ideas on how to structure the content and, in particular, how to design the web-page that researchers use to access data and documentation. The proposed outline provides, most of all, a template for the overview to be comparable across studies.

3.2 Questionnaires

Panel studies are not required to use questionnaires as instruments, but for all reference studies in section 2.2, the questionnaire is the most important type of instrument. Furthermore, the GLBPM identifies two steps of questionnaire development: the design (in phase 2), and implementation (in phase 3). The implementation is crucial because it defines how data are collected in phase 4. Therefore, this section focuses on data models for *implemented questionnaires*.

Survey methodologists usually distinguish questionnaire-based surveys regarding their *modes* of data collection. Common modes include telephone interviews, personal interviews with a printed questionnaire or with a interviewer laptop, and web-based interviews. Each mode implies various methodological issues and comprehensive knowledge about mode effects is important for every survey researcher [102]. From the perspective of the data model, however, there is only one important distinction: whether the questionnaire is printed (static) or implemented as a computer program (dynamic).

In a printed (static) questionnaire, all relevant information are visible in the layout. This includes not only questions and answer categories, but also interviewer instructions or filter definitions. The layout is fixed and cannot change during the interview situation. Printed questionnaires are usually designed using desktop publishing software (DTP) and can be archived as PDF files.

Dynamic questionnaires, implemented as computer programs, are more flexible and react to previous responses during the interview. Each execution can result in the same questionnaire being presented in a different way, depending upon how the respondent answers each question. Text elements are adjusted to previous answers, filter definitions are executed in the background hiding irrelevant questions automatically, and the layout depends on various aspects like the hardware or other software tools (e. g., the browser in the case of a web-based interview). Furthermore, the utilization of a computer facilitates new methods of data collection, like the measurement of reaction times during the interview.

Questionnaires, in particular dynamic ones, resemble computer programs in many aspects [103]. Therefore, possible representations of a questionnaire are discussed in using computer programming metaphors. These metaphors reflect the visual representation, flow logic, and data model of computer programs. (Chapter 7 introduces the Model-View-Controller pattern (MVC) [27] as a more technical example.) The same three layers can be applied to discuss possible representations of a questionnaire. Figure 3.1 illustrates the field layout (visual representation), the source code (logical representation), and the semantic model (data model) for an exemplary question.

The *field layout* (figure 3.1a) represents how the respondent (or in a personal interview, the interviewer) sees the questionnaire during interview. The layout can influence the respondent's behaviour, and it is important to document it regardless of whether it is printed or dynamic. A static questionnaire can usually be stored as a PDF file that contains all relevant information. In contrast, the layout of a dynamic questionnaire has to be recorded (for example as screen shots), but these records usually lack important information like filters or certain preloaded information used to personalize the questionnaire experience. While the layout of the static questionnaire can be considered to be sufficient for documentation purposes, dynamic ones need additional information.

Many software tools for dynamic questionnaires use some kind of program code or *source code* as an import. Figure 3.1b provides an example from NIPO ODIN [104]—a proprietary software and scripting language for computer assisted interviews. If the source code is the only obvious input, one might think that it provides a complete documentation of the questionnaire. The final representation, however, depends on various factors (e. g., hardware and software configurations) and therefore the source code cannot substitute for the documentation of the field layout. Furthermore, some tools separate the definition of a questionnaire and the template for the layout.

The *semantic model* it usually the most idealized and abstract representation of a questionnaire, containing no layout information (which might be stored in a separated layout template). In comparison to the source code, data models are usually more restricted, but these restrictions make them easier to parse or import into a documentation system. Figure 3.1c provides an example in queXML [105], a standard that is discussed in greater detail in section 4.5.

To preserve and document the content of a questionnaire, the most important representation is the field layout. In the case of a dynamic questionnaire, additional

A) FIELD LAYOUT

Which of the following is your main area of research?

- ☐ Health sciences
- ☐ Mathematics/natrual sciences
- ☐ Engineering
- ☐ Economics, social sciences
- ☐ Other (specify): [ ]

B) SOURCE CODE

```
** Q10: Disciplin ------------------------------------------------------
*QUESTION 10 *CODES L3 *LABEL "Disciplin"
*FONT 7Which of the following is your main area of research?
*FONT 0

001: Health sciences
002: Mathematics\natrual sciences
003: Engineering
004: Economics, social sciences
900: Other (specify): *OPEN *NOCON
```

C) SEMANTIC MODEL

```
<question>
  <text>Which of the following is your main area of research?</text>
  <response varName="Q10"><fixed>
    <category><label>Health sciences</label><value>001</value></category>
    <category><label>Mathematics/natrual sciences</label>
              <value>002</value></category>
    <category><label>Engineering</label><value>003</value></category>
    <category><label>Economics, social sciences</label>
              <value>004</value></category>
    <category>
      <label>Other (specify):</label><value>-oth-</value>
      <contingentQuestion varName="Q10other"><length>24</length>
      </contingentQuestion>
    </category>
  </fixed></response>
</question>
```

Figure 3.1: Three aspects of a question: the field layout is how the respondent sees the question, the source code enables the computer to render the question (example in NIPO ODIN's scripting language [104]), and the semantic model provides an abstract representation of the question (example in queXML [105]).

information in form of either the source code or the semantic model are required to document the flow logic. However, to compare the content of questionnaires or to design questionnaires for multi-mode surveys, the idealized representation in the semantic model enables further software support.

A specific problem for documentation is the flow logic. Static questionnaires usually combine filters ("Ask this question only if…") and *goto* statements ("If …, go to …"). Spencer [103] points out that *goto* statements in questionnaires are problematic for the same reasons that motivated software developers to banish them in high-level programming languages. Further, during documentation, the important information are filters that explain under which conditions a question was asked. It is very inconvenient for a researcher to search all previous questions that might contain *goto* statements influencing the current one. It is strongly recommended that, at least for the documentation, all *goto* statements are translated into filter definitions to be complete and consistent.

3.3 Research data

Panel researchers are used to the rectangular data format. Within this format, however, various designs for panel data are possible. The first part of this section presents and discusses multiple options, in particular the long format and the wide format. Furthermore, a specific challenge for panel data is the representation of missing values, discussed in the second part of this section.

In a rectangular dataset, each row represents an entity (e. g., a respondent) and each column represents a measure / variable (e. g., the answers to a particular question). Rectangular datasets are popular because many statistical models and statistical packages (e. g., Stata and SPSS) are optimized for them. Panel studies collect data in iterations (waves) and, therefore, they usually result in collections of datasets. Furthermore, most studies observe multiple entities (e. g., data on the household level and on the individual level), again multiplying the number of datasets. In this *cross-sectional* data structure with distinct datasets for waves and analysis levels, one dataset represents one class of entities (analysis unit) at one particular point in time (wave).

The panel design, however, implies that entities *and* variables recur over time. Corresponding datasets can be combined for analysis in two possible structures: the wide and the long format [56, 106]. In the *wide format* (figure 3.2a), variables from different waves are added as additional columns. Resulting in a data struc-

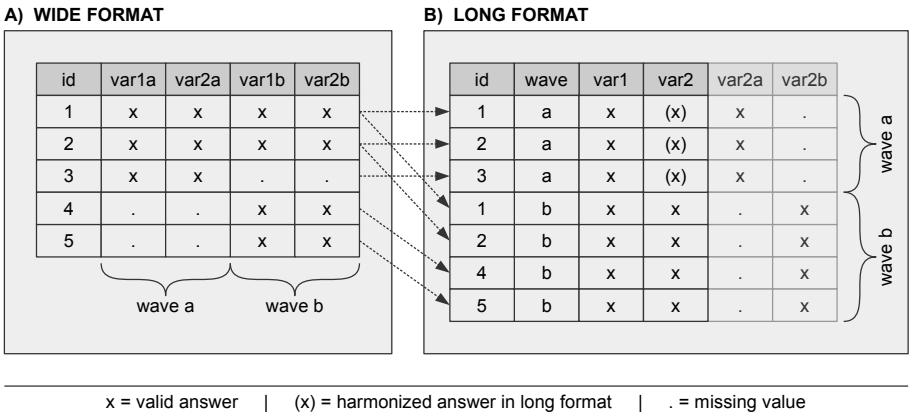

Figure 3.2: The most prominent structures for panel data are the wide and the long format. The wide format contains one row per entity (respondent), repeated measures are stored in distinct columns. In the long format, a row represents an entity at a distinct point in time. The identifier for the individual is therefore complemented with an identifier for the wave. Repeated measures are usually stored in one column; however, changes in the measure might result in a distinct columns for the original variables (`var2a` and `var2b`) and a harmonized variables (`var2`).

ture, where each row still represents one entity and each column represents one measure at a specific point in time. The most important implication of the wide format is that the formats for corresponding variables can differ over time. When transforming data into the *long format* (figure 3.2b), the researcher has to assure that corresponding measures over time are coded consistently. In contrast to the wide-format, the long format adds new rows for subsequent waves and, thereby, changes the definition of what rows represents—each row now represents one respondent at a particular point in time. Hence, corresponding measures over time can be stored in one column, but only if the measures are coded consistently. For inconsistent measures, there are two options: they can either be stored in two separate columns (`var2a` and `var2b` in figure 3.2b) or they can be harmonized to fit in one column (`var2` in figure 3.2b).

The constrain that only consistent values can be stored in one column can also be seen as one of the biggest advantages of the long format—it is obvious which variables are consistent over time and which ones have to be harmonized for analysis. These information are not directly accessible in the wide format. Therefore, the long format can be transformed into the wide format automatically but not the other way round. In computer science, the transition from wide to long could be considered as one step towards normalisation, with even more normalized structures being possible. One extreme would be the Entity-Attribute-Value model (EAV model) [107, 108], where all data are stored in only three columns (entity identifier, attribute identifier, and the actual value). Structures like the EAV model might be reasonable for data storage. For data management and analysis, the long format is considered to be the optimum.

A specific problem of research data—in particular for the social and economic sciences—are missing data originating from the interview situation. During an interview, various factors can prevent the collection of a valid response. The SOEP Desktop Companion [23], for example, distinguishes three causes: (1) The respondent refuses to answer a particular question or does not know the answer (*item non-response*); (2) the question cannot be answered by the interviewee because a necessary condition is not met (*does not apply*); and (3) a given value was found to be *implausible*.

The reasons for missing values have implications for the statistical reliability of a variable [109]. While the second cause (does not apply) simply reduces the number of valid cases, the first and the third cause (item non-response and implausibility) can also bias the results of data analysis because they interfere with

the assumption of a random sample. From a generic perspective, the answer to a question and the reason for a missing value are two different variables, but most statistical packages implement mechanisms to store the reasons for missing values in one variable together with the valid answers (which is more efficient in the terms of storage).

Stata and SPSS have implemented sophisticated functionality to deal with missing values, based on two different data structures. Stata [110, 111] has a predefined set of values for missing values starting with the `.` (dot) as the standard encoding for missing values and additional 26 values (`.a`, `.b`, ..., `.z`) for more specific missing definitions. Internally, Stata reserves the largest 27 values of each numeric data type for missing values. Because the missing values have a specific position in the data type, comparisons like `var01 < .` are possible to select the valid cases of `var01`. However, this solution works only for numeric values (including labeled numeric values) but not for characters and strings. In contrast to Stata, SPSS [112, 113] allows the user to declare one or more arbitrary values for each numeric variable as missing values. Technically, the missing declaration is part of the metadata and most statistical functions consider these declarations during analysis. Again, this mechanism works only for numeric variables.

In contrast to Stata and SPSS, R [114, 115] has no mechanism for user-defined missing values, but three pre-defined missing codes. The null pointer (`NULL`) is complemented with two additional values for non applicable (`NA`) and not a number (`NaN`). R programmers, however, found various solutions to deal with missing values but none of these implementations is considered to be a default solution for R. Duşa [116] implemented a mechanism similar to SPSS in DDIwR where every variable is complemented with an additional vector that defines which values are treated as missing values. In r2ddi [39], we[1] represent every variable in two vectors: one for the valid cases and one for the missing values (for every entity, one of these values has to be `NULL`). Nevertheless, both of these solutions are only used within one package and none of them is considered a default solution for R.

The inconsistent designs of the statistical packages motivated some studies to design their own conventions for missing values. The three missing definitions in the SOEP, mentioned earlier, are consistently coded with `-1` (no answer/don't know), `-2` (does not apply), and `-3` (implausible value). This approach works fine in SPSS, not without recoding in Stata, and has the serious disadvantage that the SOEP, by default, cannot store negative numbers in its datasets.

[1] Joint work with Jan Goebel.

In conclusion, the most flexible and efficient solution for data storage is the SPSS design. All other designs can be represented in SPSS and, therefore, it seems to be a good default for documentation purposes. As an alternative, the two-column approach in r2ddi might need more memory but has a better performance to calculate statistics and is also capable of storing paradata for valid cases.

3.4 Transformation scripts

Data transformation tasks cover all kinds of data manipulation. This includes recode operations, generation of new variables from existing ones, deletion of either variables or cases, merging or splitting data files, and statistical computations. Using statistical packages like Stata, SPSS, or R, the researchers execute these operations either manually or via script. Manual editing of data is considered to *not* be a sustainable option. Other researchers cannot track what changes are made manually to the data and the software cannot replicate the results automatically. Scripts, on the other hand, can reproduce or even replicate results, which other researchers can study to understand a particular transformation, and can be used to automate significant parts of the data processing step in the GLBPM. The introduction of metadata-driven infrastructure in chapter 4 requires that all transformation tasks are script based to enable further automation.

We can distinguish two aspects of data transformations: the *process* of performing a transformation and the *digital object* (script) that represents and executes a given transformation. Data transformations as a process will be discussed from various perspectives in the following chapters. In general, the process includes the task itself but also inputs and outputs that can connect multiple tasks. In this section, we focus on the digital object—in particular, the documentation of transformation scripts that follows the more general claim to document the code of scientific computations [10, 117].

The design of script code in Stata, SPSS, or R resembles the design of common high-level programming languages (such as C, Java, or Ruby). They work with local variables, allow the definition and execution of functions, and provide control constructs (e. g., for loops or if-then-else conditions). Furthermore, they allow users to add comments in code files that are not executed at runtime and can be used to explain in a more or less structured way the purpose and design of a script.

Based on the resemblance to software code, it seems reasonable to re-use tools and principles for software development. Version control and structured comments

are two examples. *Version control systems* (such as Git [118, 119]) manage changes and versions of scripts in a structured way and support collaborative editing using shared repositories for code. Regarding *comments*, programmers know the rule of thumb to write the same amount of comments as code. The same could apply to all transformation scripts. Tools like Sphinx [120] or Doxygen [121] can generate code documentations based on syntactic conventions for comments. Chapter 6 proposes a comprehensive reference architecture for managing transformation scripts.

The preservation and documentation of transformation scripts is tightly connected to a particular software tool. The code itself might be consumable for researchers who understand the language. The execution of the code, however, is bound to specific software. An alternative approach, proposed by the Banko Italia and Eurostat, is the design of a meta-language called the Validation & Transformation Language (VTL) [122, 123], aiming to generate reproducible representations of the algorithms that could be transformed into the scripting languages of the statistical packages. Nevertheless, exact replicability might be unrealistic because it would have to consider every single detail—even bugs—in the actual implementations of the statistical packages. Thus, even different versions of one particular package can produce different results.

Some authors [62, 124] argue that computational science should not aim for replicability but for reproducibility. In this context, a full replication would repeat all steps of a study (including data collection), whereas reproducibility would only demand enough information (including code and the original data) to get a detailed understanding of the study. If so, some problems of replicability (like the question whether updates of a particular software package might still be able to perfectly replicate previous results) might be negligible. The more important challenge is to document data transformations as a whole in a comprehensible and reproducible way. From this perspective, the VTL provides a comprehensive and valuable reference language to write reproducible but not replicable code.

The analogy to high-level programming languages will shape the following discussion of data transformations and the design of a reference architecture in chapter 6. Chapters 4 and 5 take a closer look at the possibility to automate processes using transformation scripts and the documentation of transformations as a whole (not only the scripts).

3.5 Documentation, preservation, and metadata

'Data documentation' is a collective term including a variety of information about studies and datasets that are crucial for their long-term preservation [125]. We must be precise about three terms—documentation, metadata, and preservation—as they refer to overlapping tasks and content, but seek different goals.

The *documentation* bundles material for users to understand and comprehend something. This something can be concrete objects (e. g., a dataset) or abstract objects (e. g., a study). It includes both existing material (digital objects, metadata, etc.) and additional material that is only collected or generated for the purpose of documentation.

Metadata are added to the existing digital objects to store additional information and links between objects. We define metadata as "data about data" [16]. Metadata are often considered a part of the documentation—limited to digital objects and highly structured. There are, however, metadata that are not part of the documentation (e. g., copyright protected material or temporary metadata).

Preservation is more a task than a digital object. It composes and stores digital objects (including the documentation and some metadata) to be available in the long run. Many digital objects, including the documentation and some metadata, are transformed to remain usable (e. g., by converting binary files into ASCII formats).

In analogy to the Berners-Lee's five star deployment scheme for open data [31], five levels of structuredness can be distinguished when documenting panel studies and their digital objects: prose text, structured formats, open and plain text formats, standardized formats, and linked data.

(1) The absolute minimum for a documentation is *prose text*, for example in the form of a working paper. The study description in section 3.1 can be seen as an example for a plain text documentation. Further, for very innovative instruments (section 3.2) there might be no structured documentation format available, in which case researches should at least provide a written reference. This corresponds to the first star of open data which refers to data that are available through the internet in an arbitrary format. Berners-Lee adds a second demand to this first star, which is an open license. It seems reasonable that the documentation of a study should be accessible under an open license—free and not behind the pay wall of an academic journal.

(2) *Structured formats* support computer programs to automate tasks and provide additional functionality based on these formats (e. g., a search interface). Examples are the SOEP item correspondence table [126] (providing a list of related variables in an Excel file) or proprietary formats for questionnaires. Unfortunately, those formats depend on a specific software and therefore their interoperability is limited.

(3) As an alternative to proprietary formats, structured information can also be stored in *open and plain text formats*. Examples are Comma Separated Values (CSV), the eXtensible Markup Language (XML), or the JavaScript Object Notation (JSON). Section 3.2 distinguished source code, which is usually in a proprietary format, and semantic models, which can be represented in open and plain text formats.

(4) The next step towards interoperable tools is the use of *standards*. To continue the example of a questionnaire, a standard like queXML [105] can be used to define the structure of XML files for questionnaire documentation. While the original definition of 5-star open data demands W3C standards, it seems sufficient for a panel study to use a standard for their documentation that is acknowledged by the scientific community.

(5) The fifth level of open data demands to *link data* to other data. This would also be desirable for panel data and their documentation and will be discussed in detail in chapter 5. The DDI community is currently working on an RDF version of the standard. A first draft was published as the Disco ontology [127], the next version of the standard is expected to have generic support for RDF [33].

3.6 Conclusion

The research question for this chapter asked whether it is feasible to model the digital objects, identified at the end of chapter 2, generically and independent from a specific software implementation. The discussion of the individual objects showed that there is no general answer but that every digital object comes with its own requirements and possible solutions. Nevertheless, the design of a five-star documentation provides an overall framework to work towards generic data models for all digital objects.

The minimum of a prose description was already applied in section 3.1 and its design for a study description which is only slightly structures by a group of top-level headings. Transformation scripts are usually on the level of structured but proprietary formats (level 2), but the VTL already proposed a standardized

language, which corresponds to level 4. The discussion of questionnaires distinguished the field level layout (level 1), the source code (level 2), and the semantic model (up to level 5) of a questionnaire. Similar, the formats for research datasets usually range from proprietary formats (level 2) to open standards like the Tabular Data Package (level 4). Due to the protection of sensitive microdata we usually do not aim for level 5 panel data. However, level 5 would be desirable for all kinds of related metadata. Section 4.3 will distinguish endogenous and exogenous metadata, which allows the publication of non-sensitive aggregations of microdata as linked open data.

Chapter 4

Metadata-driven infrastructure design

The history of the DDI standard illustrates a paradigm shift in the metadata communities [68]. For a long time, metadata were considered to be a part of the documentation and, accordingly, they were collected at the end of a survey project. Since version 3, the DDI standard is based on a lifecycle model (see figure 2.1). This version of the standard encourages data producers to collect and curate metadata throughout the lifecycle of the data and to reuse metadata in subsequent waves. It is supposed to increase the quality of the metadata while reducing the cost to curate them.

For many metadata experts [4, 32, 128, 129], the next step towards a more efficient design is not only to collect metadata during the entire process but to automate significant parts of the process based on metadata. National statistical institutes (NSIs), in particular, are implementing *metadata-driven* processes and infrastructures. Revilla et al. [3], for example, present a detailed use case describing the National Statistical Institute of Spain (Instituto Nacional de Estadística; INE) and its corporate-wide metadata driven production process that was designed based on the GSBPM (the NSIs' counterpart of the GLBPM). They highlight that the redesign of the technical infrastructure is required to meet the increasing demand of society for reliable data and statistics. The existing literature, however, remains vague on what the term 'metadata-driven' actually means—or is too specific, considering only one particular use case in a non-reusable manner.

Similar to previous chapters, the metadata-driven infrastructures should be designed and discussed on a generic level to support both quality and usability of panel data, while being cost-efficient. The research question for this chapter is: How can the concept of a metadata-driven infrastructure optimize the production of panel data and increase the quality and usability of the resulting data?

Chapter 1 points out that most researchers working in panel studies have no technical background. To them, the idea of a metadata-driven infrastructure might seem overwhelming as they are used to standard tools that provide limited capabilities to design highly automated workflows. It is not realistic for researchers to instantly adopt a fully metadata-driven infrastructure, or even implement it on their own. Therefore, the following sections describe a transition from what is from now on called a *document-driven* workflow to a *metadata-driven* infrastructure.

The first section takes the development of a questionnaire as an example for the transition from a *document-driven* to a *data-driven* workflow that reduces a significant amount of manual data imports. In the second section, the management of research data in the GLBPM is used as an example for a *script-driven* workflow, where most tasks are executed based on transformation scripts. The fourth section, after some background information on *metadata*, introduces the concept of a *metadata-driven* infrastructure where the transformation-scripts use not only research data as an input but also metadata. The chapter closes with an overview of *standards* to support the design of metadata-driven workflows.

4.1 Data-driven questionnaire development

Metadata-driven infrastructures aim to reduce costs in general and, in particular, transaction costs that emerge whenever a digital object is passed from one agent (researcher, software program, institution, etc.) to another. Transaction costs increase when the recipient needs to transform the incoming digital object before conducting a given task. The first example is the development of a questionnaire, including design, implementation, and documentation tasks. Figure 4.1 illustrates two possible workflows with significant differences regarding transaction costs.

In the *document-driven* process (figure 4.1a) the questionnaire is passed from the survey manager to the fieldwork manager and on to the data manager in various formats that correspond to the individual researchers' favourite tools. The survey manager might work with Microsoft Word, passing a Word file to the fieldwork manager who might use Adobe Indesign to layout the questionnaire as a PDF file,

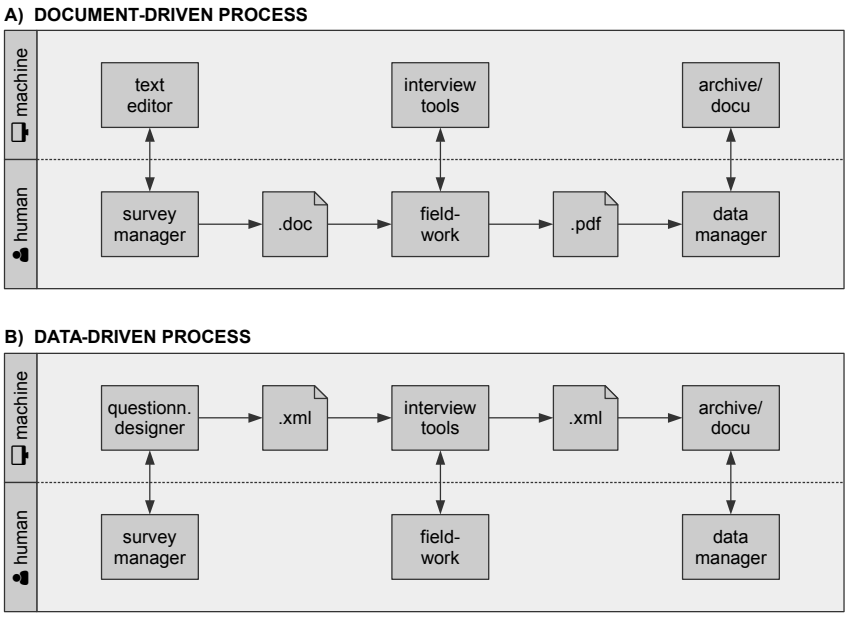

Figure 4.1: Comparison of two possible workflows for the development of questionnaires. In the document-driven workflow, the agents exchange varying data formats, resulting in additional work when the agents have to import the content manually. In the data-driven workflow, the tools can communicate directly based on a common data format.

and, finally, the data manager might use a documentation tool like Nesstar Publisher, which produces DDI-compliant XML files. Each of the three agents must perform work-intensive input and export operations—most of them manually, resulting in high transaction costs.

In contrast, a *data-driven* design (figure 4.1b) uses interoperable software products that communicate using a standardized file format (in the case of a questionnaire, XML stands to reason). In this case, the software tools can open the common format directly and no longer require the fieldwork or data manager to do any copy-and-paste work. Furthermore, the quality of the final documentation might be much better than in the first case as the potential for errors to creep in across several copy-and-paste steps is fairly high.

The transition from a document-driven to a data-driven workflow moves the responsibility for exchanging information from the human agents (survey, fieldwork, and data manager) to their software products (machines). In addition to humans and machines, there is at least one more class of agents worth mentioning: organisations. Before we continue to the next level of automation, it might be revealing to take a closer look at all three classes of agents: humans, machines, and organisations.

Humans. Fully automated tasks are independent from human inputs. Comprehensive workflows, like the nine top-level phases of the GLBPM, however, require human decisions and interventions. These processes are called *semi-automatable* [130]. The goal is to refine the description of a process to a level where every step is either *machine-actionable* or left without any further chance to identify machine-actionable sub-steps and is therefore referred to as *human-dependent*.

Machines. Many companies, like Amazon, that depend on automation have radically re-designed their infrastructure so that software systems can interoperate directly on the service level (Service Orientated Architectures, SOA) [131]. This is basically the same idea as introduced in figure 4.1b where the software systems exchange XML files. While most of the process steps of the GLBPM require human decisions, the exchange of information between the process steps and between different software systems within one process step is a good candidate for automation. In the best case, the exchange of information could happen completely automatically, bringing transaction costs close to zero.

Organisations. Figure 4.1 illustrates the exchange of information between survey, fieldwork, and data managers. For many panel studies, the fieldwork is conducted by an external organisation, adding a third class of agents: organisations.

If there is an organisational gap, the use of common file formats and technologies is most likely negotiated at the organisational level. The more organisations that are involved, the more important becomes that standards (see section 4.5) provide predefined formats.

The three classes of agents illustrate various aspects of automating data production processes. First, the amount of human interventions should be limited as much as possible. Second, software systems and infrastructures should be able to interact directly (e. g., in a service-orientated architecture). Third, standards support interoperability of software systems and are of particular interest if there is an organisational gap in the process. Nevertheless, there are limits to automating the production of panel data. The production and analysis of research data depends on the researcher's knowledge and interpretation, which prevents a complete automation of the overall process.

4.2 Script-driven data management

Figure 4.2 illustrates the flow of data in the GLBPM and identifies five typical classes of data: (1) raw data as received from the data source; (2) shared data provided to external users; (3) archived data prepared for long-term preservation; (4) analysis data on which scientific papers are based; and (5) temporary data that needs neither to be archived nor documented in detail. Each of these five classes has specific characteristics and requirements regarding data management and documentation.

The transitions between these types of research datasets can be performed based on transformation scripts (see section 3.4), as an alternative to manually editing the datasets. The use of transformation scripts has at least two advantages over manual editing. First, the scripts provide a first documentation of all transformations. Second, after preparing all scripts, the process can run automatically. This kind of automation is particularly important, when something changes in the raw data. If the data is edited manually in the first place, then all transformations must be manually corrected. Scripts, in contrast, can simply be executed again resulting in all transformations being updated automatically. Let us take a closer look at the five stages of research data in the GLBPM.

The *raw data* are the output of the data collection phase. In the example of personal interviews, raw data usually contain sensitive information at the individual level and is rarely, if ever, published. In the case of a script-driven process, raw data

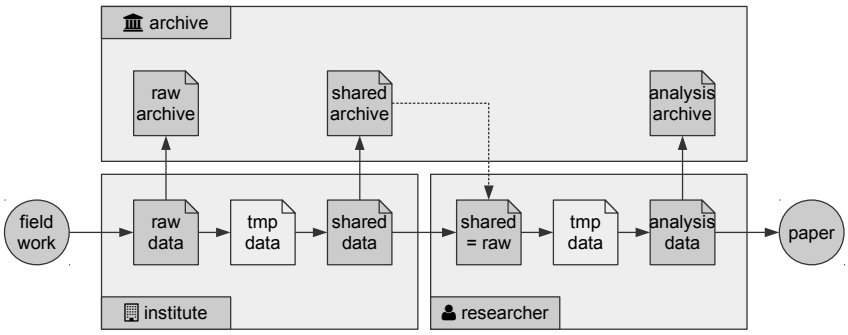

Figure 4.2: The flow of research data in the GLBPM. The image assumes three agents: a research institute processing the raw data, an archive for long-term preservation, and the individual researcher analysing the data.

are also the only class of data that is not generated based on transformation scripts. From a process perspective, however, data become raw data again whenever they are exchanged between researchers or institutions because the recipient has no access to the original transformation process. Later the shared data from the research institution becomes the raw data for researchers; as shown in figure 4.2.

The raw data are the input to the process and analysis phase, where they are prepared for publication as *shared data*. Typically, this includes tasks like validating the data quality, applying classifications on open answers, harmonising variables, and generating new variables for the users. Shared data are expected to be cleaned (inconsistent cases removed or cleaned) and user-friendly (consistent variable names, meaningful labels for variables and values, etc.).

Raw and shared data should be prepared for long-term preservation and stored in an *archive*. Shared data are usually distributed in the proprietary formats of common statistical packages because these formats are convenient for data users, but these formats are not suitable for long-term preservation because changes in the software environment make it very likely that they will not be accessible with reasonable effort in the long run. There is consensus among archivists that plain text files in ASCII format are the safest format for long-term preservation [132]. Typically, this results in a combination of Comma Separated Values (CSV) or fixed with formats for the research data and additional metadata in eXtensible Markup Language (XML) or the JavaScript Object Notation (JSON). The Tabular Data Package [133], propose by the Open Knowledge Foundation, provides a recognized stan-

dard using CSV and JSON. Section 4.5 introduces the Tabular Data Package as one possible standard for the underlying reference architecture.

After receiving shared data, most researchers run additional transformations on the data before calculating their statistical measures, resulting in a new class of data, the *analysis data*. An increasing number of journals requires that researchers publish their analysis data along with their reports. However, researchers are often not allowed to re-publish sensitive microdata. Archives and other institutions can support researchers by providing re-analysis archives (as discussed in section 1.6).

Many data transformations produce *temporary data*. These data are neither documented nor archived as long as all relevant transformation tasks are represented in respective scripts. In the terms of object orientated programming, they are very similar to private attributes and methods in class definitions that are not visible to the user of the class. Similar, the users of the shared data requires no knowledge about the temporary data.

All arrows between the types of datasets in figure 4.2 represent data exchange or transformation tasks. Thereby, transformation tasks are a specialisation of process steps in general with an input, an output and a set of rules how to transform the former into the later. To approach data transformations as process steps allows a consistent design for the given transformations. The reference architecture in chapter 6 proposes a set of rules to design these tasks consistently.

4.3 Background: metadata

The most common definition for metadata is "data about data" [16, 134, 135]. An important characteristic highlighted in this definition is the circular reference, implying that metadata are also data and can therefore have metadata describing them, thereby resulting, theoretically, in an infinite regress of metadata having metadata having metadata and so on. In the documentation of panel studies, the circular reference led to some impreciseness in definitions and discussions about metadata.

A questionnaire, for example, can be considered to be both a digital object or part of the metadata. In both cases, the questionnaire itself has metadata that include information about the author, the analysis unit, and the study. At the same time, the questionnaire could be considered to be part of the metadata describing the resulting research dataset. The digital objects, as defined in chapter 3, help us escape this problem. For this dissertation, all digital objects can be described by

metadata, but they are not considered to be metadata themselves. The questionnaire and the resulting dataset, for example, are both independent digital objects. Additional metadata can provide more details and link content across digital objects.

Ideally, the digital objects and their metadata would contain disjunct information. This would comply with the Single Source of Truth principle (SSOT) [136] that demands that every information should have *one and only one* predefined location in an information system. Many applications, however, require redundant information to perform efficiently, and the following design of a metadata-driven infrastructure is one of them.

To manage redundancies in the metadata, this dissertation proposes to strictly separate metadata that can be extracted from digital objects (*endogenous metadata*) and metadata that are completely *exogenous* to the digital object or objects they describe. For the distinction of endogenous and exogenous metadata, the only noteworthy citation is Wegman [137], who extracts endogenous metadata from text and image databases. The distinction is essential to ensure consistent data and metadata management, and, at the same time, enable automation based on metadata. A research dataset, stored as a CSV file, provides an appropriate example to illustrate the difference of endogenous and exogenous metadata, and to highlight the significance of elaborated identifiers.

Endogenous metadata describe information that can be extracted from digital objects automatically. In the example of a research dataset, this might include the dimensions of the dataset (e. g., number of rows and columns) or basic statistics for the variables in the datasets (e. g., mean, standard deviation, or frequencies). Endogenous metadata allow data portals like DDI on Rails (introduced in chapter 7) to give researchers details on a dataset without providing access to the actual data, which might contain sensitive information.

Exogenous metadata cannot be extracted from the dataset. This includes basic information, common to all digital objects (e. g., author, timestamps of creation and last modification, or usage rights) and additional information specific to a dataset (e. g., labels for variables and numeric values). A specific type of exogenous metadata are links between distinct digital objects that depend, most of all, on a stable and consistent system for *identifiers*. A variable in a research dataset, for example, is based on a particular question. Section 5.1 discusses identifiers in more detail—in particular the distinction of local and persistent identifiers.

Both endogenous and exogenous metadata are considered to be data, the prefix 'meta-' only refers to the fact that they are describing other data. Metadata can therefore be analysed like any other data to provide insights into the structure of a panel to survey methodologists, data mangers, and researchers. An example may illustrate the potentials of metadata analysis. Many questions become subject to some kind of change during the lifetime of a panel. These changes can be a result of the social reality (e. g., job market reforms) or origin from methodological issues to improve the quality of a question. In either case, metadata can help to first identify questions that are related by concept. Given a set of questions, it is simple to write tools that identify identical questions and reveal changes in the remaining ones.

4.4 Metadata-driven infrastructures

Section 4.2 introduced a script-driven design for data management where all transformations are performed based on scripts, but even this design has limitations: First, scripts usually depend on a specific software tool and its scripting language (e. g., Stata, SPSS, or R). Changes in the software environment usually requires entire scripts to be rewritten. Second, these scripts contain details that are relevant for the documentation and the understanding of the resulting data. These information, however, are coded in the specific scripting languages and are therefore hard to interpret for researchers who are not familiar with a particular software package. Furthermore, these information are usually not accessible for the documentation system.

Information hidden in the code are later documented in the metadata in order to make them available to researchers and other tools. An example is the task of renaming variables: the rename is done in a statistical package and then a table with the original and the new variable names is created to document the rename process. The question is now: If the rename is documented in the metadata, why not use the metadata as an input for the transformation script that could simply execute what is already documented? This would be the very basic idea of a *metadata-driven* design.

Metadata-driven designs are popular in various fields, including metadata communities [2, 4, 32, 129], official statistics [3, 138], and clinical database design [107, 139, 140]. Most authors, however, stay vague on what the term *metadata-driven* means to them or they define it in a very narrow context. Sundgren, for example, focuses on software products when he says that "if the operations of a software product are completely controlled by metadata, it is said to be metadata-driven"

[4]. This definition is, for example, limited to software products and the restriction 'completely controlled' is too narrow for processing research data that depends on human interpretation.

The following discussion proposes a more flexible definition for the term *metadata-driven* process that is based on the concept of *test-driven development* (TDD) [141] in software engineering. According to Maximilien and Williams, the very basic concept of test-driven development is to shift "from unit test *after* implementing to unit test *before* implementing" [142]. Test-driven development is proven to increase the quality of software products [142, 143, 144, 145].

Accordingly, in a metadata-driven infrastructure, metadata are collected *before* the corresponding objects are created or processed. Metadata can then be used to execute and control a particular task. Unlike Sundgren's definition, it seems not necessary to create tasks that are completely controlled by metadata—it might still be reasonable to complement the metadata with additional scripts and other inputs. In general, the more the process is controlled by metadata, the less additional work is required to document it.

From a technical perspective, metadata can be integrated in a script-driven process in two ways: as an additional input to transformation scripts, or by generating transformation scripts from metadata using a preprocessor. Continuing the example of renaming variables in a dataset, the original Stata file (figure 4.3a) is bound to Stata as the statistical package and is not reusable in other packages. The CSV file in figure 4.3b provides an alternative, storing the old and new variable names in a two-column design. The CSV file can be used as an input either to the final script or to a preprocessor. Figure 4.3c implements the first solution in R, importing the CSV file and iterating over its entries to perform the actual task of renaming variables. Figure 4.3d, on the contrary, uses Ruby as a pre-processor to generate and execute Stata code. Regarding the benefits of a metadata-driven design, it is worth mentioning that even in a non-metadata-driven design, it would still be expected to generate the CSV file accompanying any kind of code as part of the documentation.

In addition to this very basic example, many tasks identified in the GLBPM can be supported or automated using metadata. Table definitions (describing research datasets) can indicate identifiers (both primary and foreign keys), variables that are expected to change over time, and other metadata. Those metadata can be used to automate the validation of datasets. Variable and value labels can be extracted from the semantic model of the questionnaire, combined with a linkage file from question identifiers to variable names. The documentation of the data and the study

A) Stata script: rename.do

```
use input/data.dta
rename var1 var1a
rename var2 var2a
rename var3 var2b
rename var4 var2c
rename var5 var3a
rename var6 var3b
save output/data.dta
```

B) Metadata in CSV format: rename.csv

```
org,des
var1,var1a
var2,var2a
var3,var2b
var4,var2c
var5,var3a
var6,var3b
```

C) R script utilizing the metadata: rename.R

```
data   <- read.csv("input/data.csv")
rename <- read.csv("metadata/rename.csv")
for(i in nrow(rename)){
  names(data)[rename[i, "org"]] <- rename[i, "des"]
}
write.csv(data, "output/data.csv")
```

D) Ruby as a preprocessor for Stata: prepare_rename.rb

```
rename = CSV.read "metadata/rename.csv", headers: true
stata_code = "use input/data.dta\n"
CSV.each do |row|
  stata_code << "rename #{row.to_hash["org"]} #{row.to_hash["des"]}\n"
end
stata_code << "save output/data.dta"
```

Figure 4.3: Metadata-driven design for renaming variables

can be generated fully automated based on an comprehensive set of metadata. And these are only few examples.

4.5 Standards

Automating processes involves various technical standards for data, software, hardware, and networking. Examples are protocols like TCP/IP or data formats like HTML or XML. The following discussion focuses on standards that are used to describe the digital objects in a panel study; table 4.1 gives an overview of appropriate standards. This list might not be exhaustive but presents at least one option for each process step and digital object.

The most generic standard for metadata might be the *Dublin Core Metadata Element Set* [148], defining only 15 elements like identifier, creator, language, description, and title. In cases where no other standard is available, one could at least use these 15 elements to provide a basic set of metadata. Since 2009, Dublin Core is official endorsed by the International Organisation for Standardization (ISO) as ISO 15836 [162]. We can consider Dublin Core as a default solution that also complements other standards that otherwise neglect the basic information covered in Dublin Core.

The *Data Documentation Initiative* (DDI, introduced in section 2.1) currently provides two complementary versions of its standard: the DDI Codebook standard (version 2) [98] and the DDI Lifecycle standard (version 3) [19, 20]. DDI Codebook is a narrow standard for the documentation of datasets in a codebook format. This includes some related information about the study or the instruments, but the focus lies on the datasets. In contrast, DDI Lifecycle aims to support and map the whole process (lifecycle) of a study. This version of the standard is designed broadly, covering 846 elements in 21 namespaces [163] (in comparison, for example, to the Dublin Core Metadata Element Set that covers only 15 elements).

Beyond standards like Dublin Core and DDI, which cover a broad range of use cases, there are also more specific standards for most of the digital objects. The following takes a closer look at standards for the study description, questionnaires, research data, transformation scripts, and the documentation of a study.

There are currently no noteworthy stand-alone standards for the *study description* available, but comprehensive standards like DDI cover important aspects. Section 3.1 proposes an outline for a study description based on DDI Codebook el-

Table 4.1: Selection of standard that cover the process steps in the GLBPM and the list of digital objects.

Name	Description
DDI Codebook (DDI-C)	Narrow standard for the description of (research) datasets in a codebook format [98]. An even smaller version is available in the DDI-Lite standard [100].
DDI Lifecycle (DDI-L)	Comprehensive standard for the social, economic, and behavioural sciences [19, 20, 146]. Covers most digital objects from a metadata perspective.
Digital Object Identifier (DOI)	Published as ISO 26324:2012, the DOI system provides an infrastructure for the registration and use of persistent identifiers [147]. Further details and alternatives are discussed in section 5.1.
Dublin Core	General purpose metadata standard for digital objects. Works as a default solution for all digital objects [148].
Generic Statistical Information Model (GSIM)	Similar to the discussion in chapters 2 and 3, GSIM is designed as a generic information model [149]. Standards like DDI and SDMX are considered implementations of this model
ISO 11179	Storing organisational metadata in a metadata repository [150, 151, 152, 153, 154, 155]. The standard considers, in particular, the institutional aspects of metadata management.
Open Archival Information System (OAIS)	The OAIS standard considers both humans and systems to be part of an archive that preserves information and makes them accessible [156].
queXML	XML-based standard for questionnaires [105], supported by Limesurvey and interoperable with the DDI standard.
Statistical Data and Metadata eXchange (SDMX)	The SDMX initiative fosters standards for the exchange of metadata and statistical information with a focus on financial data [157]. DDI and SDMX are complementary standards [158].
Tabular Data Package	The Tabular Data Package provides an ASCII format for rectangular datasets stored as CSV (data) and JSON (descriptive metadata) [133]. The standard reuses other standards [159, 160, 161].
Validation & Transformation Language (VTL)	Software-independent syntax for the documentation of data validation and transformation tasks [122, 123].

ements and recommends using prose text rather than a fine structured metadata standard to ease the documentation of innovative designs and new methods.

Regarding the representation of questionnaires, section 3.2 distinguishes three aspects: the field layout, the source code, and the semantic model. The field layout can be captured in various layout formats like PDF, JPEG, or PNG. The source code solely depends on the tools used for data collection, there is no standard for questionnaire source code available. In contrast, there are various standards available for semantic models, including the DDI standard, the Questionnaire Editing & Deployment Markup Language (QEDML) [164], the Quest Markup Language (QML) [165], queXML [105, 166], and others. The queXML standard is used in the reference architecture because it is supported by LimeSurvey and interoperable with the DDI standard [167].

The five classes of *research data* (see figure 4.2) can be divided into formats for archiving (archived data) and formats for data processing and analysis (raw, temporary, shared, and analysis data). For data archiving, ASCII formats are considered to be the best option for long-term preservation because they are independent for specific software implementations that might change over time [132]. The *Tabular Data Package* [133] proposed by the Open Knowledge Foundation (OKFN) combines two established formats: Comma Separated Values (CSV) for the research data and JavaScript Object Notation (JSON) for additional metadata essential for the use of the research data. The design includes three other OKFN standards: Data Packages [159], JSON Table Schema [160], and CSV Dialect Description Format [161]. Even if these formats are suitable for all tasks at hand, most researchers rather use the more convenient *proprietary formats* of their preferred statistical packages (e. g., `.dta` files for Stata or `.sav` files for SPSS) to process and analyse the data.

The format for data transformation scripts depends, most of all, on the software in use. These are either statistical packages (like Stata, SPSS, and R) or database management systems (like relational databases using SQL as a query language). The previous section on metadata-driven designs suggests storing the definitions for transformations in the metadata, only using scripts to process them. To become independent of the proprietary formats, Banko Italia proposes a first draft for a generic Validation & Transformation Language (VTL) [122, 123], introduced in section 3.4.

While there are various standards for metadata, there are no considerable standards for the study documentation. Many research institutions and archives, how-

ever, provide useful guidelines (e. g., the UK Data Archive [42, 125], the ICPSR [132], and the Digital Curation Centre [168]). Corti et al. [125] suggest a two-part documentation consisting of a study-level and a data-level documentation. The study-level documentation corresponds to the digital object of a study description. The fact that the second part of the documentation is at the data-level serves to highlight that the data are the most important 'product' of a panel study and their description is often the core of the documentation. Structured information, in particular metadata, are important for building a comprehensive documentation system. However, accompanying material like PDF questionnaires also provide valuable insights.

4.6 Conclusion

In section 4.3 we distinguish between endogenous and exogenous metadata. The former describes metadata that can be extracted automatically from a digital object, the later refers to metadata that are distinct from the digital object, provide additional information on the content, or link multiple objects. The distinction of endogenous and exogenous metadata provides the foundation to implement metadata-driven infrastructures without breaking with the Single Source of Truth principle.

The term 'metadata-driven' is defined with an analogy to test-driven development in software engineering: The traditional way of managing metadata considered it as part of the documentation and therefore metadata are usually collected after the respective digital objects are created or processed. In a metadata-driven infrastructure, the metadata are prepared before the respective task, such that data transformations and other tasks can use them as an input. This is more efficient, helps to create software-independent workflows, and is more error-resistant.

During the discussion, the main application was the data generation process, but the development of questionnaires and other digital objects could be optimized in a similar way. The four steps from document-driven to metadata-driven, with data-driven and script-driven designs as interim stages, are valid for other use cases as well. Even the management of prose text documents, like the study description, can be optimized this way—using, for example, markup languages like Markdown (introduced in more detail in chapter 6) to automate the production of various documentation formats.

The term ‘metadata-driven’ is defined in analogy to test-driven software development, and while the focus lies on automation, metadata could also be used to generate tests for research data and other digital objects. The transition from metadata-driven to test-driven data production is a subject for future research. The distinction of endogenous and exogenous metadata might a valuable input for such a discussion because endogenous metadata could be used to define expectations for subsequent waves.

Chapter 5

Variable linkage

Data from household panels can be combined in various ways to be analysed, the most common cases compare related measures over time and integrate different levels of analysis (e. g., enriching data on the individual level with data from the household level). Furthermore, it is popular to combine panel data with other data sources. Merging multiple data sources is distinguished in two categories, based on the links to combine the data: record linkage [8] uses a common identifier to link identical entities whereas statistical matching [5] links similar entities based on a common set of attributes.

All of these examples and techniques—whether combining data from one panel or merging multiple data sources—come with one work intensive challenge: to find corresponding variables across multiple datasets and data sources [6]. This can be corresponding variables over time, a common set of attributes for statistical matching, or variables that are in at least one of exactly two sources for any kind of data merging. In addition to the requirements of data merging, a second type of relations between datasets and data sources results from data transformation tasks, which link input and output variables. Section 3.4 already discusses how to document transformation scripts and their execution—missing is a design to document the relation between the inputs and outputs. While record linkage and statistical matching simply require an omnidirectional relation, the documentation of data transformations represents a unidirectional relationship.

In a metadata-driven design—as the previous chapter proposes—it stands to reason that these relationships are stored in the metadata. This would not only provide the necessary information, but would also enable software tools to provide

additional support (e. g., finding corresponding data sets based on a common set of attributes). This section takes a closer look at possible data structures, asking: Which information about related variables do survey researchers require to analyse distributed data sources (e. g., panel data, statistical matching, or record linkage) and how can they be covered in the metadata?

The chapter starts with a general discussion of identifiers, which are a crucial part of any representation of links and relations between variables. Before discussing existing solutions, we take a closer look at possible use cases for variable linkage. Existing solutions are either implemented by panel studies or suggested by standards like DDI. Two solutions stand at reason for the problems described above: concepts for omnidirectional relations, and a process-orientated documentation for unidirectional transformation tasks.

5.1 Background: identifier systems

Introductions to identifier systems usually present persistent and globally unique identifiers as the most important class of identifiers [169, 170, 171], which are also important for the design of a metadata-driven infrastructure and to link research data on the variable level. During data processing and similar tasks, however, local identifiers (or 'internally unique identifiers' [172]) are equally important. It would, for example, be preposterous to replace a variable name like `age` (local identifier) with a globally unique identifier like `7fc6d1e6-a2e6-11e4-89d3-123b93f75cba` (UUID). At the same time, it is unlikely for a local identifier like `age` to be unique in the context of multiple studies.

Richards et al. [172] distinguish globally unique and persistent identifiers, implying two dimensions for the discussion of identifiers: *scope* (global versus local) and *accessibility* (persistent versus temporary/not persistent). Scope refers to the context in which an identifier is unique. Local identifiers (like variable names) depend on a specific context to assure uniqueness, while global identifiers (like UUIDs) are expected to be unique without further context definitions. Accessibility considers the relation of an identifier to the object that it refers to. Some identifier systems (like DOIs) are designed to provide a persistent link from any identifier in use to the related objects. Other systems (like URIs) do not assure that a given identifier is linked to anything at all. Based on the two dimensions, the following discussion distinguishes three kinds of identifiers: local identifiers, global but not persistent identifiers, as well as global and persistent identifiers.

Local identifiers depend on their context and can be optimized for this context. In a research dataset, (variable) names provide local identifiers for columns. They can be optimized to be easy to use and easy to understand, while Universal Unique Identifiers (UUIDs) and other globally unique identifiers would be overwhelming, as demonstrated before. In other contexts, sequential numbers provide useful local identifiers—questions in a questionnaire are usually numbered consecutively and relational database systems suggest sequential numbers to generate primary keys automatically.

The most popular system for *global but not persistent identifiers* are Uniform Resource Identifiers (URIs) [173] and the aforementioned UUIDs [174]. URIs are the identifier system used by the world wide web. In contrast to Uniform Resource Locators (URL), which are used to identify a particular resource on the internet, URIs can be used to identify almost anything. Semantic Web technologies, for example, reference and link objects even if they are not represented on the Internet [175]. UUIDs provide an alternative system based on random numbers. Because of the size of the numbers (128 bit) and the system to generate them, the probability of a duplicate is negligible and UUIDs are considered to be globally unique. Even if URIs and UUIDs are globally unique, they do not assure that a given ID can be resolved to find a particular object.

Global and persistent identifiers, like Digital Object Identifiers (DOI) [147] or the DDI identifiers system [19, 176], take precautions to assure that a given identifier is actually linked to the related object. Both systems are based on a two-step registration system where (1) the provider assigns registration agencies and (2) every registration agency can then register persistent identifiers within its namespace [177]. Unlike URIs that can identify basically anything (even physical objects), DOIs are restricted to digital objects (like publications, data, or visualizations) and DDI identifiers to metadata elements compliant to the DDI standard.

Each of these systems has its advantages and disadvantages. Local identifiers are convenient but have a very limited scope. Global but not persistent identifiers are easy to use but the lack of persistence limits their use. Global and persistent identifiers are technically the best solution, but many researchers feel uneasy that the providers might vanish or misuse their virtual monopoly and they are inconvenient for many use cases.

The following solution is proposed for metadata-driven infrastructures: During all processing steps, local identifiers are used to reduce complexity. There is no need to use global and persistent identifiers for work in progress that includes temporary

Table 5.1: Use cases for variable linkage within one study (internal) or combining multiple data sources (external). The last two columns indicate which metadata design is appropriate for the respective use case: concepts as a common reference (Con.) or the representation as a transformation step (Tran.).

		Relation	**Con.**	**Tran.**
internal	1	related by content	X	
and	2	related by constructs	X	(X)
external	3	related through transformation	(X)	X
	4	variables related to instruments		X
	5	variables related to other material	X	X
	6	relations facilitating metadata-driven designs		X
external	7	re-analysis	X	
only	8	aggregated data	X	
	9	statistical matching	X	
	10	record linkage	X	
	11	data citation		X
	12	interoperability / standards	X	X

objects. When digital objects and metadata are either archived or published, UUIDs are assigned as globally unique identifiers. These UUIDs can be converted into global and persistent IDs by combining them with the namespace of an registration agency (which is possible for global and persistent identifiers that use a two-step registration system as mentioned above).

5.2 Use cases for variable linkage

Digital objects in a panel study are related in various ways, some of which are of particular interest, like the link from a variable to the instrument or the links between variables that are based on repeated measures. Table 5.1 condenses the results from the SOEP user survey, the development of DDI on Rails, and the earlier discussion on metadata-driven infrastructures to a list of twelve use cases of relations amongst digital objects.

(1) Repeated measures over time create a special demand for finding related variables in a panel study. As long as the repeated measures are consistent, the use case is relatively simple. Unfortunately, measures in panel studies are often updated over time, either for methodological reasons or due to changes in the

observed reality. Simple changes are, for example, updates of a question text or adding an additional item to a single-choice question. The use case gets more complex, when the number of resulting variables varies—for example, because a single-choice question was transformed into a multiple-choice question or the other way round. In this first use case, the researcher is looking for *variables related by their content*.

(2) The second use case links *variables related by constructs* like psychological scales. The Big Five personality traits are, for example, represented in the SOEP questionnaire with 16 items that can be transformed into five dimensions [178]. The documentation is expected to group both the 16 items (this use case) and the resulting scale variables (next use case).

(3) Data transformation tasks (see sections 3.4 and 5.5) create a relationship between input and output variables. The minimum requirement for the documentation is to link *variables related through transformation*. A more sophisticated documentation would include details about the transformation itself.

(4) Variables are not only based on transformations, but in most cases, they are based on instruments. *Linking variables to instruments* concerns both the direct relation from raw variables to the instruments and the transitive relations of variables that are based on transformations but are still highly related to the original design of the instrument (e. g., the results from one question after data cleaning or in a harmonized long variable).

(5) In addition to the links to other variables or instruments, variables can be related to all kinds of material (e. g., publications or methodological documentations). Use case 5 asks for generic solutions to link *variables to other material*. The nature of this material is not necessarily known up front.

(6) Chapter 4 proposes a metadata-driven design for data processing and illustrated the idea with the example of renaming variables based on metadata. This is basically an extension of use case 3 with one additional requirement: *Relations facilitating metadata-driven designs* require the metadata to be available before the actual transformation is executed [2].

The first six use cases link variables within the scope of one study, but they are also valid when linking multiple data sources. Multiple data sources, however, provide additional challenges that, usually, do not apply within a single study.

(7) Researchers like Egloff et al. [179] use different data sources to verify existing results with new data, in which case the entities do not have to match. To enable re-analysis of previous results with new data, these new data have to provide the

same set of information as the original data. If related variables are not only linked within one study (use case 1) but across studies, possible data sources for *re-analysis* can be identified solely on the basis of their metadata.

(8) The first use case for merging data sources is to enrich given microdata with *aggregated data*. We find various examples where the SOEP data are combined with external data sources. The weights [180], for example, are based on data from the official statistics and geographical data are enriched with indicators from the Microm dataset [181].

(9) In *statistical matching* [5], a common set of variables (see use case 7) is required to match similar cases and, afterwards, additional variables are combined for analysis. This use case consists of two parts: the identification of variables that are common to all relevant data sources (common attributes for the matching algorithm) and the identification of variables that are available in at least one of the relevant data sources (the distributed attributes to be combined).

(10) *Record linkage* is also intended to combine data sources with different sets of information, but now the data share a common identifier for the individuals (entities). It has become popular to link panel data with administrative data—SHARE, for example, links to the German Pension Fund (DRV) [182], while the SOEP drew an additional migration sample in collaboration with the German Employment Agency in 2014 [183]. The project Processes of Mate Choice in Online-Dating (PPOK) linked process generated data from an e-dating platform with additional interviews of its users in a panel survey [7].

(11) In figure 4.2, the shared data from the data producer become the raw data to the work of the researcher (data user). To keep the whole process comprehensible, the researcher would have to link his work back to the data source. *Data citation* not only acknowledges data producers [11, 35] but is also a requirement that enables reproducible research [184, 185].

(12) The last three use cases combine metadata from different data sources. Unfortunately, the design of panel data and their metadata usually does not consider the combination with external data sources—a phenomenon, often referred to as 'data silos' [186, 187]. Furthermore, innovative instruments like the Implicit Association Tests (IAT) [70] or biomarkers [188, 189] result in more complex data structures than standardized questionnaires. To ease the combination of metadata a common data model is desirable. The last use cases demands *interoperability*: the solution should be generic enough to work with established standards and data

models. This includes the combination of data and metadata in different data formats (e. g., CSV, XML, or JSON.).

5.3 Existing solutions for panel data

Existing panel studies and the DDI standard offer various solutions to document the relationships between variables in the metadata. The SOEP, for example, keeps a record of related variables in a spreadsheet format (the 'item correspondence list') [23, 126], while the GIP and SHARE use time-consistent identifiers within its variable names [190, 191]. The DDI standard offers at least four feasible solutions [19, 20]: direct links, variable groups, concepts, and versioning.

As of 2015, the design of the SOEP documentation is under heavy development; the following illustration refers the traditional design: The 'item correspondence list' [126] links related variables over time. This document and the related documentation system SOEPinfo [51] are of particular relevance for the SOEP data users because the variable names are designed to change over time, even if the variables are based on identical measures—the variable name is a combination of three elements: wave identifier, questionnaire identifier, and number of the question/item. The item correspondence list is designed as a rectangular dataset where columns represent waves, and rows represent sets of related variables (called 'items'). This design has various problems (e. g., that only one variable per wave and item is possible), but nevertheless, it covers use case 1. In SOEPinfo the item correspondence list is complemented with links to the questionnaires to fulfil use case 5.

Other panel studies designed conventions for *time-consistent variable names* (e. g., the naming conventions of the German Internet Panel [190] or the Release Guide of the SHARE Project [191] that also consider versioning on the variable level). In general, identifiers are not considered to be part of the metadata and should not contain any information other than the identification reference [94]. From the users perspective, however, it is convenient if the variable names are consistent over time, facilitating use case 1.

The DDI Lifecycle standard [19, 20] covers various designs to link variables. Unfortunately, multiple solutions to one problem do not only increase the flexibility of the standard but also provide a new problem: to select the best solution for the given context. Hansen et al. [93] distinguish two basic approaches to link variables and data in DDI: implicit and explicit data comparison techniques. Implicit techniques utilize concepts or DDI resource packages to link variables. Explicit tech-

niques involve groups, versioning mechanisms, and the DDI comparison module. These techniques blend generic data models with technical details of DDI as an XML standard. At this point, we are only interested in the generic model and not the technical implementation. The given examples cover four generic designs to link variables: directly linking and comparing two variables, grouping variables, linking variables to a common concept, and versioning of a given variable.

Direct links (DDI: comparison module) provide a solution for the simple comparison of two variables (e. g., representing a recode operation). However, many comparisons and, in particular, data transformations link more than just two variables (e. g., a transformation task might require multiple inputs) [93]. Direct links are a good solution to compare two versions of one variable (e. g., to document a variable in a panel has changed from one year to another), or to document a simple recode operation. Section 5.5 proposes a more generic solution for data transformations that can be seen as an extension of direct links with multiple inputs and outputs. The comparison of two variables then becomes a specialisation of the more generic solution.

Variable groups, including the recently introduced `RepresentedVariable` element [192], are an efficient design to link related variables within one panel study while being capable of documenting changes over time [193]. Nevertheless, group definitions from multiple studies are unlikely to be interoperable unless they have been harmonized—there are too many possibilities to structure such groups. Due to the limitations when documenting multiple studies, the idea of variable groups is not considered to be a sustainable solution. The idea to use a grouping mechanism to document the differences in related elements, however, seems promising and will be taken up in the following section.

The idea to use *concepts* to link variables is very similar to the idea of variable groups. From the perspective of a data modeller, both solutions follow the principles of a hub-and-spoke architecture where one central element (the hub) is used to link a group of related elements (the spokes) in the most efficient way [194, 195]. Groups and concepts differ, most of all, regarding the interpretation of the hub. Variable groups are more technical, representing the design of variables and instruments, whereas concepts are more general and focus on the meaning of variables. Two variables representing the gender of a respondent might be split into two different groups of variables if the answer categories are coded differently (e. g., the first variable codes 0 = male, 1 = female and the second variable codes 0 = female, 1 = male), but this difference would not be sufficient to assign a different concept.

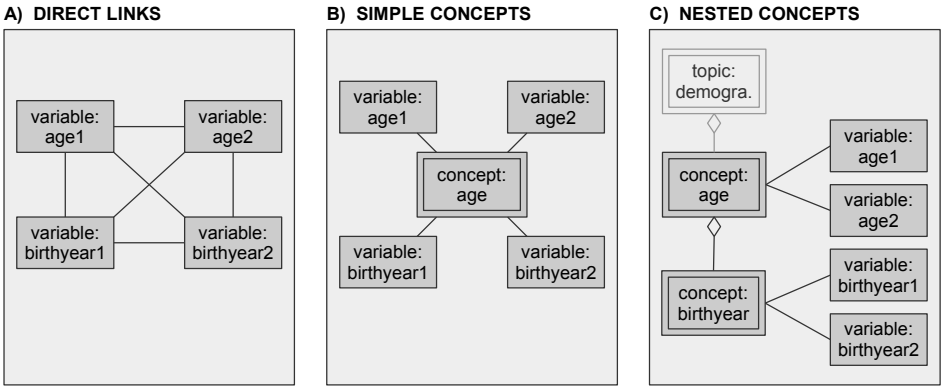

Figure 5.1: Concepts are a more efficient design than direct links to represent related objects. A) The number of direct links grows exponentially with the number of objects. B) Using concepts, on the other side, requires only one link per object. C) Furthermore, concepts can be nested or otherwise structured to create a more powerful documentation.

Versioning of variables can complement other solutions like direct links or concepts [93]. In DDI, multiple versions of an object still have the same identifier—an additional version number is used to distinguish them. In the context of metadata, it is crucial to distinguish what is versioned: the actual object or the metadata describing it. As an example, the content of a variable can change (new version of the object) or the variable label can be updated (new version of the metadata element).

5.4 Concepts, statistical matching, and record linkage

Data from one single study are usually structured consistently, whereas the combination of data from multiple sources is very likely to confront researchers with diverse or even conflicting data structures and designs. The simple and generic design of concepts, linking data elements based on their content, provides a flexible approach to link within *and* across data sources. Basically all objects from chapter 3 can be linked to concepts, especially variables, questions, and publications. And even if objects are not identical but related, concepts can be used to link those objects and identify differences automatically.

ISO 11179 defines a *concept* in part II of the standard (Metadata Registries: Classification) as a "unit of knowledge created by a unique combination of characteristics" [151]. The definition is referenced and reused by the DDI standard [196]. A *unit of knowledge* can represent both specific objects (e. g., a respondent, a building, or a car) and abstract ideas (e. g., happiness, love, or satisfaction.). Concrete and abstract concepts correspond to aspects of reality investigated by researchers. Thus, concepts provide not only a flexible mechanism to link variables, but can also support researchers in finding variables based on their content.

Concepts can be structured hierarchically to represent relationships and to ease the navigation within a given set of concepts. Thesauri, tables of contents, or the SOEP topic list are typical examples. Semantic Web technologies and the idea of linked open data provide more elaborated data structures and technologies to link content on a *conceptual* level [197, 198]. Metadata portals like DDI on Rails can furthermore use concepts to integrate thesauri into their search interfaces.

The DDI standard nests concepts in concept groups [196], resulting in an hierarchical structure that corresponds to common designs for thesauri or tables of contents. Unfortunately, the given data model has limitations for non-atomic concepts that can be differentiated further on. In surveys, this might happen when a single-choice question is split into a multiple-choice question or the other way round. In this case the group of variables that result from the multiple-choice question are considered to be sub-elements of the variable resulting from the single-choice question. To represent this structure on the conceptual level, concepts are required to be nestable.

The DDI-related Discovery Vocabulary (Disco) [127] does not take its definition for concepts from the ISO 11179 standard but reuses the Simple Knowledge Organization System (SKOS) [199], where concepts can be nested in a broader/narrower relationship. Figure 5.1c illustrates an alternative data-model with nested concepts.

Concept groups and nested concepts serve different purposes. Concepts groups are similar to a table of contents and define an overall structure for concepts whereas nested concepts represent substantial variations of a given concept (not technical variations). DDI on Rails uses both designs: Concepts are nested and reused amongst studies to provide interoperable metadata, while concept groups (called 'topics') might become study specific. This would allow studies to structure their respective variables and other digital objects individually.

A common list of concepts across multiple studies provides valuable information to users but causes new problems for participating studies. The researchers

implementing those studies are used to care about consistent data and unique identifiers within the scope of one study. The new design requires consistent concepts and persistent identifiers amongst multiple studies. This can be achieved by using study namespaces, implementing concept registries, or linking concepts as linked open data [175].

The role of *namespaces* is similar to the role of registration agencies for persistent identifiers (see section 5.1): concept identifiers only have to be unique within a particular namespace. They ensure that one concept does not refer to conflicting objects. However, they cannot prevent that one object might have multiple concepts referring to it.

A centralized *concept registry* would ensure the uniqueness of concept identifiers and supports researchers who assign concepts in searching for existing definitions of a given concept. A concept registry provides a technical solution as a common namespace for multiple studies. Thus, namespaces and concept registries are considered to be complementing solutions. If, however, more than one concept registry exists, each one would have its own namespace and concept registries could no longer assure that there are no redundant concept definitions and identifiers.

Semantic Web technologies and especially the idea of linked open data provide an elaborated framework and reusable technologies to implement a decentralized infrastructure to manage and utilize concepts [30]. Linked open data are represented as graphs (triples) and can link multiple notes that represent the same concept through a 'same as' relationship. This provides an alternative solution to the problem of multiple identifiers amongst namespaces where it is no longer necessary to eliminate duplicates but to connect them.

The three solutions (namespaces, concept registries, and graphs) complement each other and can be combined. First, every study could assign concepts within its namespace. Second, studies could collaborate in curating a common set of concepts. Third, existing concepts can be harmonised *ex post* as linked open data. On the first two levels (namespaces and concept registries), the proposed data structure combines concept groups (DDI 3.2) with nested concepts (SKOS). This provides a sophisticated framework to represent even complex concepts while the hierarchical structure of concept groups is easy to comprehend. The SKOS concepts are already bases on an ontology suitable for linked open data and therefore prepare the third step of linking concepts *ex post*.

Concepts are a powerful tool to link variables. The given design supports at least 8 of 12 use cases as highlighted in table 5.1, including both internal and exter-

nal links amongst variables and other digital objects. The general nature of links based on concepts is omnidirectional, for example to group similar variables. It is not suited to represent directional relations as they occur in data transformation processes.

5.5 Data transformation as a process

Section 3.4 illustrates the problems of documenting transformation scripts. Even so, the goal of a perfectly reproducible documentation seems almost impossible and sometimes doubted [124]. Documentation should, at minimum, provide enough details to use the generated (output) data correctly. Continuing the discussion of script-driven and metadata-driven data processing (sections 4.2 and 4.4), data transformation is seen as a process step with an input and an output as well as some task in between.

In computer science, the black-box and white-box metaphors describe two basic principles to discuss process steps (such as data transformations). A black-box design completely hides the functional principles of a tasks from the user and tells only what inputs are allowed and what outputs are expected. The white-box design, on the other hand, gives complete insights into the functional principles itself. Furthermore, some authors suggest a grey-box metaphor as a pragmatic middle ground [200, 201]. We can discuss and document data transformation tasks using all three options.

In a *black-box* documentation, a simple relation between the input and output variables helps researchers to find related variables. A short note on how input and output variables are related might be desirable, but is not mandatory. As an indirect reference, the output variable's label usually gives some indication of the transformation purpose. The SOEP documentation of the generated datasets `pgen` [202] and `hgen` [203] provides an extensive and recognized example of a black-box documentation for generated data.

In contrast, a *white-box* documentation reveals detailed insights on the transformation process. The simplest version includes the respective fragment of the script code that performed the transformation. Code fragments are, however, often hard to comprehend for the human reader and sometimes require a significant amount of context information generated at runtime. Generic languages, like the Validation & Transformation Language (VTL) introduced in section 3.4, can be used to docu-

ment simple transformations in a replicable manner, but they are expected to fail for more complex examples [204].

The most valuable approach might be a compromise of these first two options. A *gray-box* documentation could combine a prose description of a transformation with some kind of pseudo-code that illustrates the inside of the box without being replicable. The goal is to provide a comprehensible, but not exhaustive, documentation of the transformation. The grey-box approach significantly relaxes the expectations regarding a generic language like the VTL, which makes it realistic to use such a language for documentation. The documentation has to be comprehensible for a researcher to understand the transformation, but it has no longer to be exactly replicable.

A simple data model for the documentation of data transformations, considering the grey-box metaphor, could be based on the process model of the OWL-S ontology [205] which is designed as a semantic markup for web services but can be easily reused in the given context. Besides inputs and outputs, the model also knows *locals* which can be used to represent metadata used in a metadata-driven process as proposed in section 4.4. Further, the original OWL-S model is supplemented with a human readable description and a field for pseudo code. The design of a reference architecture in the following chapter considers the OWL-S model.

The documentation of unidirectional data transformations complements the omnidirectional design of concepts. The OWL-S ontology provides a process-orientated design for the documentation of data transformations. The process model is quite generic and can be reused for other unidirectional processes, for example, to document the relation of questions to variables.

5.6 Discussion

This chapter asks what information about related variables is required to analyse distributed data sources (e. g., panel data, statistical matching, or record linkage) and how it can be incorporated into metadata? After a detailed introduction on identifier systems and a discussion of existing solutions in panel studies and the DDI standard, the discussion proposes two solutions to document relations between variables: omnidirectional links based on concepts, and a process-orientated design for unidirectional links.

Concepts enable us to link variables not just within one panel study over time but also across multiple studies and data sources. A concept represents a ‘unit of

knowledge' created by a unique combination of characteristics [151]. Concepts are designed more generic than other data models (such as variable groups or the DDI comparison module), which becomes advantageous when linking data from different sources or formats because the design is not tied to a specific data model. Furthermore, concepts provide the necessary link to find interoperable data sources for more complex data merging techniques like record linkage or statistical matching.

The process-orientated approach to document data transformation tasks is more flexible than the traditional designs that we find in panel studies or the DDI Lifecycle standard. First, it allows different levels of documentation detail, including black-box designs (where the functional principles are hidden), white-box designs (where the functional principles are reproducible), and grey-box designs, which are not exactly reproducible but comprehensible for other researchers. Second, unlike other data models for direct links (like the DDI comparison module), process models (like OWL-S) support multiple inputs, outputs, and even local variables—providing a powerful framework for the documentation of transformation tasks and other unidirectional relationships.

Concepts and data transformations can support automation following the discussion in chapter 4. Concepts, in particular, support the automation of tests in panel studies. Based on common concepts, related variables can be identified for automated comparison. Further work might ask whether it is possible to use statistical tests to design acceptance tests to test new variables. T-tests could be used to identify variables where the mean in a new wave differs significantly from previous years. The simplest test for categorical variables would look at the number of categories, their labels, and the frequencies for the categories. If it is possible to automate many of the tests, this would be additional prove that concepts are sufficient and do not need to be complemented by panel-specific metadata like variable groups.

Part II

Proof of Concept and Application

Chapter 6

Reference architecture

Part 1 designs a theoretical framework for conducting panel studies, automating data processing, and enhancing documentation. In particular, chapter 4 designs an infrastructure for metadata-driven processes and chapter 5 discusses various data models for linking data on the variable level to enable data merging techniques including record linkage and statistical matching. So far this discussion is on abstract and theoretical.

This chapter proposes a reference architecture of a metadata-driven infrastructure for panel studies with two aims. It provides researchers with a defined set of designs and principles to work towards a metadata-driven infrastructure; and it establishes a software environment for further development projects, including the design and implementation of DDI on Rails. It starts with a set of file formats (based on the standards in section 4.5) and selects a corresponding tool suite that is entirely based on open source tools and, therefore, easy to replicate (see table 6.1 for an overview). To design script-driven or metadata-driven workflows for data processing, a standard directory layout provides a basic structure for all process steps covered in figure 4.2 on page 60. A set of fictitious test data enables developers to test new designs without touching sensitive information included in the actual research datasets of recognized panel studies. The chapter concludes with a set of design patterns that are mostly based on more general principles for software development.

Table 6.1: Reference architecture: File formats and software tools for the digital objects. All tools are open source examples.

Digital object	**File format**	**Tool suite**
study description (phases 1, 2, and 9)	output-independent markup (Markdown in various dialects)	wiki or content management system (Gollum wiki)
questionnaire (phases 3 and 4)	data model (queXML) field layout (PDF and PNG), and source code (PHP templates)	questionnaire designer and survey tool (LimeSurvey)
research data (phases 4–8)	data files (Tabular Data Package)	statistical software (R)
transformation scripts (phases 5–8)	scripts (R scripts)	version control and build tools (git, Github, and Rake)
documentation (phases 6 and 7)	metadata storage (CSV or relational database for exogenous metadata; and XML for endogenous metadata)	generic data editing tool (LibreOffice Calc or Base); and the R package r2ddi
publication (phase 8)	markup language (Markdown, alternative: LaTeX)	typesetting and additional tools (Pandoc and R, alternative: LaTeX, knitR, R, and BibTeX)

6.1 File formats

Section 3.1 designs the study description as a prose text document where only the top-level structure is pre-defined. *Markdown*, originally proposed by John Gruber [206], provides a basic set of formatting commands to describe text documents in a output-independent format. Thus, a study description in Markdown can be converted into HTML (for a web-based documentation system), print-optimized PDF, and other formats. Special dialects like Scholarly Markdown [207], Github Flavored Markdown [208], or R Markdown [209] provide additional functionality for researchers. R Markdown, for example, can be used to thoroughly document R scripts or, the other way round, to embed R code and its results into scientific publications.

Section 3.2 distinguishes the semantic model, the field layout, and the source code of a questionnaire. The semantic model is represented in *queXML* [105], an XML standard for questionnaires that is supported by LimeSurvey and is interoperable with the DDI standard. The field layout is documented either as screenshots in the PNG format or as a print layout in PDF. LimeSurvey does not represent a questionnaire's content in a source code format. The web representation, however, uses PHP templates for rendering, which are considered to be source code that must be documented in order to facilitate the replication of the original tool.

Statistical packages have their own binary formats, but these formats are, at best, partly interoperable, with most undergoing significant changes over time. This makes them neither suitable for centralized data storage nor for long-term preservation. The *Tabular Data Package* [133] format (introduced in section 4.5) provides a plain text (ASCII) solution that also provides a sustainable format for long-term preservation. To keep it simple in the reference architecture, we use the Tabular Data Package for both data processing and data archiving tasks.

The reference architecture uses R for data processing and analysis. Out of the three preferred packages identified in section 1.5, R is the only open source tool. Using open source tools ensures that the reference architecture is reproducible for other researchers without further costs. It implies that all transformation scripts are stored as *R scripts* as well. *R Markdown* [209] is used to add detailed comments into the code files. Google [210] and Wickham [211] provide detailed style guides to structure and format R scripts that are easy to reuse and to comprehend for other researchers.

The discussion of user requirements in section 1.3 identifies a preference for rectangular data formats over hierarchical ones. The DDI standard, however, is based on XML and is not suited to be mapped directly to any rectangular format (such as CSV or relational databases) [163]. For the development of DDI on Rails, a DDI-based data model for *rectangular data structures* was designed [40]. Based on this data model, a set of tables is defined that can be managed either in CSV files or in a relational database. The CSV files cover exogenous metadata, which have to be edited by data managers. For exogenous metadata, which are automatically extracted from the research datasets using the R package r2ddi, the DDI Codebook standard (XML) is still appropriate.

Previous chapters did not discuss the generation of scientific publications. To provide, however, an end-to-end example for all major GLBPM process steps from design to evaluation, tools for generating a small report are included. They demonstrate that even this step can be designed completely script based for the purpose of partial automation and complete reproducibility. Again, R Markdown provides a simple solution to write scientific papers tightly bound to R code. A more sophisticated solution might combine R and LaTeX through the knitR package [212].

6.2 Tool suite

Except for the transformation scripts, which are tied to the software R, the file formats are software independent. The list of tools is considered a proof of concept that it is actually possible to implement a reference architecture using these formats and the design patterns presented afterwards. However, there are plenty of alternatives for every suggested product. The following tools are open source tools or free services for which open source alternatives are available. The constraint on free tools ensures that the whole reference architecture is reproducible without additional cost for software licences. Further, the tools focus on two core technologies—Git for version control and Markdown as the common markup language—to create a consistent selection.

Git [118] is a version control system with a decentralized design [213]. It works both locally, on a researchers computer, and remotely, as a repository to enable collaboration. Basically, Git can store all file types, but it is optimized for plain text files like R scripts or Markdown documents. Binary formats and large files are usually excluded from version control and are stored on file servers. The open source repository tool Gitlab [214] and its proprietary alternative Github [215] can

be used to host and manage Git repositories. Github is free for open source projects and is a *de facto* standard for open source projects—it is therefore used to host the test case and DDI on Rails.

The reference architecture stores Markdown files (e. g., containing a study description) in a *Gollum wiki* [216]. Gollum uses Git to store the content in plain-text files. Furthermore, there are complementary tools available: the site generator Jekyll [217] produces static HTML websites based on Markdown, and the converter Pandoc [218] converts Markdown files into many other markup languages (including LaTeX, MediaWiki markup, and HTML) or binary formats (including PDF, Microsoft Word dotx, and LibreOffice Writer ODT). Gitlab and Github both complement repositories with Gollum wikis.

Data management and analysis is done in *R* [219]. Unlike Stata and SPSS, which are domain specific software tools for social, economic, and behavioural researchers (SPSS stands for 'Statistical Package for the Social Sciences'), R is a software environment for statistical computing *and* a programming language. Chambers [61] highlights that the design of languages like R enables researchers to become programmers who create R packages and thereby provide new functionality. There are currently more than 6,000 R packages available [220]. In the reference architecture, all steps described in section 4.2 are conducted using R to ensure a fully script-driven (and in some parts even metadata-driven) workflow. Furthermore, R is used to develop additional functionality that is required but not yet available.

Markdown uses triple backticks (```) to indicate code junks. R Markdown [209] is both an R package and an extension to the original definition of the Markdown language. The intention is to first execute the R code embedded in the file and after that render the result. The modifications in R Markdown allow to define, in more detail, how embedded R code is executed and what to do with the results. Researchers can, for example, embed code to generate a visualisation that will be included in the final rendering of the file as a PDF document.

The open source software LimeSurvey [59] is used as a questionnaire designer and a collection tool for web-based interviews. LimeSurvey supports queXML [105] as an exchange format, providing also a print version of the questionnaire in PDF format. queXML covers the semantic model of the questionnaire. Screenshots or the PDF version can be used to document the field layout. For reproduction, the template for the screen version is stored as PHP source code that can be used to reproduce the full version of the questionnaire. The resulting data can be exported in CSV format with supporting R or SPSS scripts for the first processing step.

LibreOffice Calc [221] provides a simple editor for CSV files, which are used to store metadata. Script-based manipulation of those CSV files can be done in R. CSV files are plain text and can be managed in Git. Nevertheless, the use of CSV has its downsides: LibreOffice Calc and R have different defaults for storing string cells (R always puts strings in quotes whereas LibreOffice only quotes if necessary), and common diff tools (such as the `git diff` command) are optimized for changes in columns (lines), not for changes in columns. Again, to keep the reference architecture simple, CSV files are sufficient and we can ignore the problems. For larger studies, however, it is recommended to use a more sophisticated solution for metadata like a relational database (e. g., PostgreSQL [222]).

This set of tools provides us with the basics to implement a metadata-driven infrastructure and to automate significant parts of the whole process. The purpose of this chapter is to illustrate the concepts from the first part of the dissertation and to prepare an environment for the introduction on DDI on Rails in the following chapter. It does not aim to provide an exhaustive catalogue for software tools. For illustrative reasons, we focus on simple and seasoned tools. In large panel studies, however, more sophisticated tools and technologies might be required. In particular the storage of files, data, and metadata might be supported by various tools for data management and data exchange. Relational databases and NoSQL technologies (including various tools for cloud storage) supplement the Git repository from the tool suite. Semantic Web technologies provide more sophisticated technologies to publish and utilize data and metadata. Build tools, like Rake for Ruby programmers or the original GNU Make, are used to optimize more sophisticated process steps—for example, to control a workflow combining Stata, R, LaTeX, and other tools. In addition to these general purpose tools, scientific workflow management systems (e. g., Kepler or Taverna) provide specialized tools for research data management.

6.3 Standard directory layout

The steps from phase 4 (data collection) to phase 8 (research and publication) in the GLBPM can be designed to run completely script-based. This implies a fully replicable design where every single step is at least documented by the script that executes it. Figure 4.2 (page 60) illustrates the relationships between the different stages of research data. The arrows are process steps that transform one or more

input datasets into one or more output datasets. If all of these transformations are completely script based, the entire process can be run fully automatically.

Script-based transformations are basically process steps with inputs, outputs, and the scripts performing the transformation (see section 5.5). In consideration of this basic design, the following discussion develops a standard directory layout for transformation tasks, integrating them into the reference architecture. A standard directory layout defines a set of directories and files, and complements it with rules (design patterns) for the organisation of the process.

Standard directory layouts are used in various programming languages and frameworks including R [223], Apache Maven [224], and Ruby on Rails [225]. Familiarity with the standard directory layout makes it easy for developers or researchers to structure new projects or to become familiar with existing projects. The proposed structure in this section considers FritzJohn's design [226] for R projects, McCullough's recommendations for an effective archive [185], and the previously mentioned standards for R packages (scripting language), Maven (compiled language), and Ruby on Rails (framework). The resulting layout consists of a minimum of four directories and two files, but can be complemented with additional directories as required (see figure 6.1).

This directory layout is optimized to work with Git, dividing the directories and files in two groups: The first group includes inputs, outputs, and temporary files which all should be ignored by the version control system. The second group represents the details about the process step (including scripts and their documentation)—version control is used to create backups and enable collaboration.

We ignore the `input/` and `output/` directories in version control. This highlights the separation of data as one class of digital objects and transformation scripts as another class of digital objects that process data (see chapter 3). The standard directory layout is optimized to manage and document transformation scripts and not the datasets that are processed—the later are therefore ignored in version control. The input directory might contain large quantities of files and a process might also generate multiple and complex outputs. The two directories usually contain further sub-directories like `data/`, `metadata/`, or `documentation/`, but these sub-directories are not pre-defined by the standard directory layout.

The whole transformation process can be initiated by executing a single file, the `main.*` file. The asterisk (`*`) indicates that the standard directory layout does not restrict the file format or the language of the main file. The use of R in the reference

```
project/
|-- docs/     # Human-readable documentation (complementing readme)
|-- input/    # Input files (not in version control)
|-- output/   # Output files (not in version control)
|-- scripts/  # Scripts, producing output out of the input
|-- main.*    # Executes all scripts in the correct order
|-- readme.*  # Explains the purpose/structure of the project
|
|##### OPTIONAL #####
|
|-- temp/     # Temporary files (not in version control)
|-- meta/     # Additional metadata
|-- local/    # Local (system-specific) parameters
|-- import.*  # Definition of how to fill the import folder
|-- lib/      # Classes (script language)
|-- src/      # Source code (compiled language)
|-- bin/      # Binaries (compiled language)
|-- test/     # Tests (unit, integration, and regression)
|-- test.*    # Run all tests
```

Figure 6.1: The standard directory layout for data transformation tasks consists, by default, of four directories and two files. The default can be complemented with additional files or directories as necessary.

architecture suggests a `main.R` file. However, shell scripts (`main.sh`), Windows batch files (`main.bat`), Ruby scripts (`main.rb`), and various other languages would do the same job as long as the execution of this one file initiates all tasks of the transformation. The existence of one single file that executes everything else ensures two things: First, all programs in the `scripts/` directory are executed in the correct order. Second, the whole transformation does not need further user inputs and is therefore considered to be automated.

Most transformation tasks are too complex to be programmed as one single file. The `scripts/` directory stores additional scripts which have to be initialized by the `main.*` file. This design also allows researchers to combine multiple languages in one transformation. A shell script (`main.sh`), for example, can execute R scripts, Stata do-files, or LaTeX files stored in the `scripts/` directory in one iteration.

The `main.*` file and its complementing `scripts/` directory ensure that the transformation is machine actionable. The same way, the `readme.*` file and the `docs/` directory ensure that the content of the transformation can be understood by human researchers. The `readme.*` file is the starting point that refers to any further files stored in the `docs/` directory, which might also contain material like PDF questionnaires or related publications. And again, the asterisk indicates that the file format of the readme file is not determined here. Nevertheless, to be in line with the rest of the reference architecture, Markdown (`readme.md`) is recommended.

These four directories and two files are not exhaustive. Tools like Stata sometimes require temporary files (`temp/`) or local variables (`local/`), which are neither inputs nor outputs, but should also be ignored by version control. In a metadata-driven design, an additional `metadata/` directory might complement the existing design to store metadata that are prepared as part of the particular process step and should, therefore, be under version control. Furthermore, for a chained set of projects, an `import.*` file can define how to fill the `input/` directory if the inputs depend on previous steps and might change.

An increasing number of researchers are developing new tools to extend existing functionality [13, 61]. Those development projects usually start as part of a given transformation or analysis tasks and are later extracted into a distinct package. Scripting languages (like Ruby or R) can store class definitions or functions in a `lib/` directory. Compiled languages (like Java) separate the source code `src/` from the compiled binaries `bin/` (the later should be excluded from version control).

Software tests are a common practice amongst developers. Software tests cover single units of code (unit tests), code compositions (integration tests), or the over-

all behaviour of a program (regression tests) [13, 141]. Like software developers, researchers can also test their transformation scripts by writing tests. The design of tests, however, raises certain requirements regarding the code to be tested. Languages like R that can structure code in functions are easier to test than pure procedural tools. Furthermore, data managers can complement tests for their scripts with test for the data in `test/`.

6.4 Test Data

The file formats (section 6.1), the tool suite (section 6.2), and the standard directory layout (section 6.3) enable us to implement a basic infrastructure for data production, management, and analysis. To test this infrastructure, however, we need a test case. The SOEP data and other microdata contain sensitive information and are only available under certain restrictions, which would also apply when using them as test data. Thus, we[1] created a fictitious panel study including non-sensitive test data to provide a use case that is available without any restrictions. Unlike randomly generated data, which would also contain no sensitive information, the fictitious test study reproduces specific characteristics of a panel study including intentionally placed errors in the data to make the use case more realistic.

The test study represents a household panel with a sample of 20 individuals in 10 households and three waves of data collection. The fictitious data are based on a small screen-play for the respondents to ensure consistent data over time. The instruments for each wave include a household and an individual questionnaire for all respondents and an additional biographical questionnaire for new participants. The questionnaires are implemented in LimeSurvey, and the questions are slightly modified over time to simulate the development of a study. The use case is small by intention, so that all transformations from the raw data to the final import in DDI on Rails run in a couple of minutes on a personal computer.

6.5 Design patterns

The implementation of a metadata-driven infrastructure requires detailed knowledge about programming and software infrastructures. Most researchers, however, are at best self-taught programmers—lacking knowledge on how to structure soft-

[1] Joint work with Carolin Stolpe and Linda Zhu

ware code. Wilson et al. [13] propose a list of common practices for software development to be used by researchers. Their practices are in line with the previous discussion, including a focus on human-readable and reusable code, robust and fault tolerant programs, automation, refactoring, as well as the use of collaboration tools. In conclusion to the reference architecture, I would like to highlight three of their best practices (collaborative code development, embedded documentation, and 'plan for mistakes') and add five additional practices that are more specific to data management for larger projects.

Collaborative code development suggests, first of all, establishing a review process for code before using it in production. This includes pair programming as a more sophisticated solution, where two coders sit in front of one computer: one focusing on the details and the other on the high-level view. Furthermore, collaboration tools like issue trackers and code repositories support collaboration and increase the quality of the resulting code.

The *embedded documentation* principle suggests, in particular, to document the design and the purpose of code, not the mechanics. This is a crucial extension to the general expectation that code should be documented. Many researchers and less experienced programmers document *how* a particular piece of code works. The more important information, however, would be *why* the code was written in the first place. Additional information on how it works are only required for exceptionally complex code junks, which are rare in processing and analysis scripts. Technologies like R Markdown [209] or documentation systems like Doxygen [121] allow to extract comments that are embedded in code to generate a user friendly documentation.

Plan for mistakes suggests to write tests for the code. Those tests can ensure that particular parts of the code work correctly (unit test, e. g., testing that a recoding procedure produces correct results). Furthermore, integration tests ensure that the whole program (or in the terms of the GLBPM, a particular process step) works correctly, and regression tests check that the program still works with modified parameters.

Besides these more general practices concerning software development and all kinds of scientific computing, there are some practices that are of particular relevance for managing panel data: Single Source of Truth, Software as a Service, research data in the long format, metadata first, and plain text (ASCII) formats.

The *Single Source of Truth* principle [136] demands that every information should have one—and only one—defined place to be stored. This avoids redundancies

and inconsistencies in the data that could emerge from concurrent manipulations of the same information stored in different locations. A very common example for a violation of this principle is the management of variable labels in Stata. First, a researcher labels a variable using a script which contains label definitions. Next, running the script produces a Stata data-file containing the labels. Finally, a data manager extracts the labels and adds them to the documentation system. Which of these locations (script, data file, or documentation system) provides the single source of truth? The metadata-driven infrastructure (proposed in chapter 4) would solve the problem by generating the metadata with all labels first. In this case, the script would no longer contain any label definitions but would use the metadata input. And the labels in the Stata dataset would be considered a view on the metadata. The single source of truth, however, would be the metadata.

Collaboration issues can often be solved using web applications (like the questionnaire designer Qlib, the data repository CKAN, or Google Docs for collaborative editing), applying the *Software as a Service* (SaaS) principle [26]. SaaS applications often have only one or a very limited number of installations. The users access the service that runs online via the Internet. This principle contrasts with traditional desktop applications that must be installed on the local computer of the user. Software as a Service ensures that there is a single source of truth for the data and that all researches work with the same tools to manage the data to keep the database consistent. Furthermore, SaaS applications are usually easier to manage and update for the developers of the system.

When it comes to data formats, the preferred design for panel data is the *long format* where subsequent waves are appended as new rows [56]. Section 3.3 already argued that the long format is the optimum for panel data. It forces data managers to care about the consistency of the data and harmonise the data if necessary. Furthermore, the long format can always be transformed into the wide format automatically, which is not true the other way round.

The concept of a metadata-driven infrastructure is introduced in chapter 4. It highlights that most metadata could already be captured before the digital objects they describe are processed or even generated. By applying the *metadata first* principle, metadata can be utilized to simplify and improve the generation of the actual data product.

Raymond [227] argues that text streams are the simplest solution to be interpreted by a variety of tools. Similarly, many archives consider *plain text data formats* (e. g., CSV stored as ASCII files) to be the most stable format for long-term preser-

vation [132]. Another example are generic markup languages (e. g., Markdown) for text documents. In comparison to proprietary and binary formats (e. g., Stata's `dta`-files for research data or Word's `docx`-files for text documents), ASCII-based formats can be opened with any text editor, imports and exports are much easier to implement, plus collaboration can be supported with version control.

Chapter 7

DDI on Rails

Data providers use a variety of tools to present and document their data online. A popular example is the Microdata Cataloging Tool (NADA) [228], provided by the International Household Survey Network (IHSN). NADA implements a subset of DDI Codebook that can be prepared in the Nesstar Publisher [229]. NADA is used for the World Bank's Central Microdata Catalog [230] and the Open Metadata Survey Catalog [231]. Popular alternatives include the Colectica toolsuite [232, 233], which is based on DDI Lifecycle and provides a comprehensive set of tools to manage metadata across the data lifecycle, and CentERdata's Questasy [234, 235], which is used for the documentation of the Dutch LISS panel. Further tools are available from the DDI tools catalogue [236]. Unfortunately, none of these tools are suitable to document panel studies like the SOEP in a generic way. NADA and Colectica lack support for the specific characteristics of a panel. Questasy is used to document panels (in particular the LISS panel) but lacks support for variable linkage as described in chapter 5.

DDI on Rails is designed to be a generic tool to document panel studies. The original goal was to supersede SOEPinfo [51], reproducing its functionality but with generic support for multiple panel studies. The list of requirements includes four core requirements: First, DDI on Rails should document the specific characteristics of a panel study. Second, it should be study-independent, using the DDI standard in order to be re-usable for other studies. Third, one installation should be able to document multiple studies and multiple versions of one study. Fourth, the data users should be supported with additional functionality, like search interfaces, variable baskets, and script generators.

The design of DDI on Rails covers exactly one process step in the GLBPM—data dissemination and discovery. In the beginning of the project, the design included capturing metadata along the lifecycle, and the database was intended to be the central place to store metadata (following the Single Source of Truth principle). This might have worked for one particular study (in this case the SOEP) but it would have caused serious problems when including external studies like Pairfam and the GIP. Adjusting the design to fit exactly one process step (and one process step only), reduced the requirements for participating studies. The imports are mostly based on CSV files which cover a sub-set of DDI Lifecycle. However, the system does not set any restrictions on how to generate the import files—accordingly, there are also no restrictions on how to manage metadata. This is very important to keep in mind: DDI on Rails is a data portal supporting data users, designed as a redundant view on data and metadata. It is not the single source of truth for internal data and metadata management, but the access point for data users to obtain, understand, and use the data and metadata.

The following introduction of DDI on Rails starts with the software architecture, which is based on tools like Ruby on Rails and related web technologies, the DDI standard for metadata, and the statistical package R. Section 7.2 discusses four aspects of the data model: the use of the DDI standard, the logical level, the use of identifiers, and the implementation of the data model. The actual functionality is split in two parts: the user functionality for researchers and the interoperability with other systems. The chapter closes with an outlook regarding further developments of DDI on Rails. The documentation can be found on the project's website [40]. The software is used in production on paneldata.org [41].

7.1 Software architecture

DDI on Rails is Software as a Service (SaaS) designed, where most of the application runs on a server [26]. The (human) user accesses the software in a web browser. Other programs (like R) can use the Application Programming Interface (API) to access metadata directly. The name 'DDI on Rails' implies the two core technologies: Ruby on Rails provides the framework to implement a SaaS application, and the DDI standard is used as a framework to design the data structure of the internal database and the imports/exports. Furthermore, the web application is complemented with an additional R package 'r2ddi,' which extracts endogenous metadata from data files. The whole technology stack is optimized for the reference

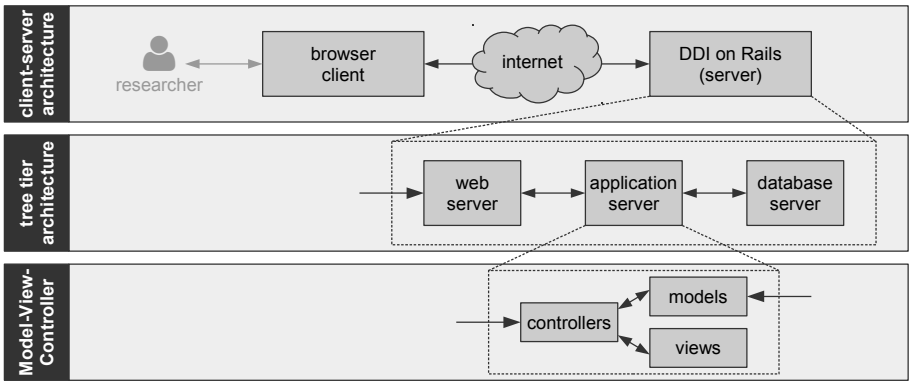

Figure 7.1: SaaS from a bird's eye view (based on Fox and Patterson [26]).

architecture that was proposed in chapter 6. Nevertheless, the reference architecture provides a recommendation for the use of DDI on Rails, not a requirement.

Ruby on Rails follows established design patterns for web applications, principally the Model-View-Controller pattern (MVC), the implementation of RESTful interfaces, and Convention over Configuration (CoC) to implement Software as a Service applications (SaaS) [26]. Figure 7.1 gives an high-level view on DDI on Rails as an SaaS application and how it is implemented. Internally, the server is divided into three tiers: (1) the web server provides an interface to the internet; (2) database server(s) run a relational database as a primary data storage and an additional search index; and (3) the application server hosts DDI on Rails. The three tiers can run on one or more machines—it is more of a logical than a physical differentiation. Internally, the application follows the MVC pattern, which separates classes that control the application logic (controllers) from the data (models) and the rendering of the user interface (views) [27].

The user interface utilizes the Bootstrap CSS framework [237] and two JavaScript libraries: jQuery [238] and D3 [239]. Bootstrap provides a general framework for the user interface. It supports responsive design techniques that optimize the interface of DDI on Rails to work on touch screens and even smart phones. The support for smart phones was more of a gimmick in the beginning, but it turned out to be an important feature for some researchers, who started checking for possible variables on their smart phones during meetings or while at conferences. jQuery, currently one of the most popular JavaScript frameworks, is part of both Ruby on Rails and

Bootstrap to provide basic client-site functionality. D3 is a JavaScript framework for interactive visualizations, used to create basic charts for the frequencies of categorical variables.

The results from the SOEP user survey indicated that the variable search is the most important functionality for researchers (see table 1.1 on page 20). The relational databases (either SQLite [240] or PostgreSQL [222]), which is used as the primary data storage, is therefore complemented by an inverted search index. The index is based on Apache Lucene [241], using Apache Solr [242] as a business layer. Inverted indices are mostly used to implement efficient full-text search. In the case of DDI on Rails, however, the index also solves another problem: Due to the complex relations in the data model, search requests often require details from at least a dozen tables with nested relations. Apache Solr is capable of storing this information as annotated fields, which are also used to create aggregates (sometimes called 'facets') to improve the search interface.

DDI on Rails documents both exogenous and endogenous metadata. Endogenous metadata originate in particular from research datasets, including variable statistics and even variable / value labels (in the case of Stata or SPSS files). We[1] implement an R package that extracts endogenous metadata from research datasets. The first version of *r2ddi* supports Stata datasets and exports DDI-C compliant XML files. Furthermore, we are preparing imports for SPSS files and the Tabular Data Package, as well as exports to JSON and CSV. r2ddi is based on the R packages `foreign` to import binary formats, `XML` to build the XML export, and `parallel` to support multicore processing.

The internal data model of r2ddi is based on the DDI Codebook standard. Again, it is not possible to map the XML-based DDI standard to the internal data structures of R directly, but the problems were less severe than for the mapping from DDI Lifecycle to a relational database because R can represent hierarchical structures and the DDI Codebook has a simpler, mostly hierarchical structure. In the terms of the Hub and Spoke metaphor [195], the internal data model represent the hub that all imports are mapping to and that all exports are build upon (spokes). Objects in R are designed similar to JavaScript objects. Future versions of r2ddi and DDI on Rails will therefore switch from XML to the JavaScript Object Notation (JSON) as an exchange format.

The biggest challenge when implementing such a converter for research data is the representation of missing values, as described in section 3.3. First, R does

[1] Joint work with Jan Goebel.

not support missing values like other statistical packages. Second, the ways that Stata and SPSS deal with missing values fundamentally differs, such that neither can represent the missing value design of the other without the risk of conflicts. As suggested in section 3.3, a two-column approach can represent valid and missing values side by side, without losing information or creating conflicts. It is also more efficient for generating statistics, which usually distinguish valid and invalid cases.

The separation of r2ddi and DDI on Rails ensures that no sensitive information is imported into the web application. The standard XML export in r2ddi aggregates all data into basic statistics (e. g., the number of valid cases, the mean for numeric variables, and frequencies for categorical variables). These statistics are considered to be completely anonymized, containing no sensitive information at the individual level. Thus, DDI on Rails does not store any sensitive information, which is important because the application has to run on a web-server. Servers containing sensitive microdata usually have higher security standards and should not be connected to the Internet.

7.2 Data model

This section takes a closer look at four aspects of the model: the use of the DDI standard and its mapping to the relational database, the logical level as an extension to the concepts introduced in section 5.4, the use of three kinds of identifiers to optimize DDI on Rails in different contexts, and the implementation of the data model in the database and the exchange formats.

The primary reference for the design of the data model is the DDI standard, both the DDI Codebook and the DDI Lifecycle standards. Because DDI Codebook is more concise, it is preferred as an exchange format (e. g., between r2ddi and DDI on Rails). The comprehensive DDI Lifecycle standard, on the other hand, provides the framework for the internal data structure and additional CSV formats for metadata imports. The XML-based standard had to be modified to fit into a relational data structure [163], which is the main reason that DDI Lifecycle is sometimes considered to be a framework rather than a standard.

The DDI Alliance publishes the standard as an XML-standard—in former years in form of Document Type Definitions (DTD) [98], now as XML Schema Definitions (XSD) [243]. Research institutions, however, tend to use relational databases to manage data and metadata. Ruby on Rails is also optimized for relational databases as a primary storage. Amin et al. [163] discuss challenges and possible solutions

when using DDI with relational databases. The problem is recognized by the DDI Alliance—the next version of the standard will still include an XSD-based definition of the standard, but it will be based on a abstract model [33]. Currently, the plan is to define the model in the *Unified Modelling Language* (UML) [244, 245]. Based on XMI as an intermediate format, various representations of the standard could be generated, including RDF representations and schema definitions for relational databases. However, as of 2015, this is a work in progress and there is no suitable option for the development of DDI on Rails. In reusing concepts from the DDI 3.2 standard to design the database behind DDI on Rails, the design will hopefully adopt easily to the next version of the standard.

One of the most important aspects of the DDI standard implemented in DDI on Rails is the idea of *concepts*. Variables and other materials are associated with concepts to facilitate linking them over time and across studies. In a panel study, however, not only the measures are surveyed repeatedly but the resulting variables are also published repeatedly, resulting in multiple versions of one variable (surveyed at one point in time). The design of concepts in section 5.4 does not intentionally include a versioning mechanism. The data users, however, expect versioning details to be presented and, in particular, to be used to lead them to the latest version of a variable. The following approach is based on the separation of the logical and the physical representation of a variable in the DDI Lifecycle standard.

DDI on Rails separates three levels of variable documentation, illustrated in figure 7.2: the actual physical representation, the logical level that links multiple versions, and common concepts on the conceptual level. Based on these three levels, variables can be queried, compared, and linked in various ways. The *physical level* represents a specific version of the shared data as the user might access them. Even datasets that date back a couple of years might change from time to time. A recent example in Germany is the 2011 census that will result in updated weights for all waves of the SOEP. Persistent identifiers are used on the physical level to capture such differences in the data. The *logical level* provides the link between different versions of one datasets. Furthermore, the logical level can store information that are common for all versions—like references to questions and concepts. Linking each version of a variable to the underlying question would cause redundancies and unnecessary work for the data manager. The logical link also enables tools like DDI on Rails to identify differences between multiple versions of one variable. The *conceptual level* is study-independent. As described in section 5.4, concepts are intended as links to the real world. At the same time, the conceptual level allows

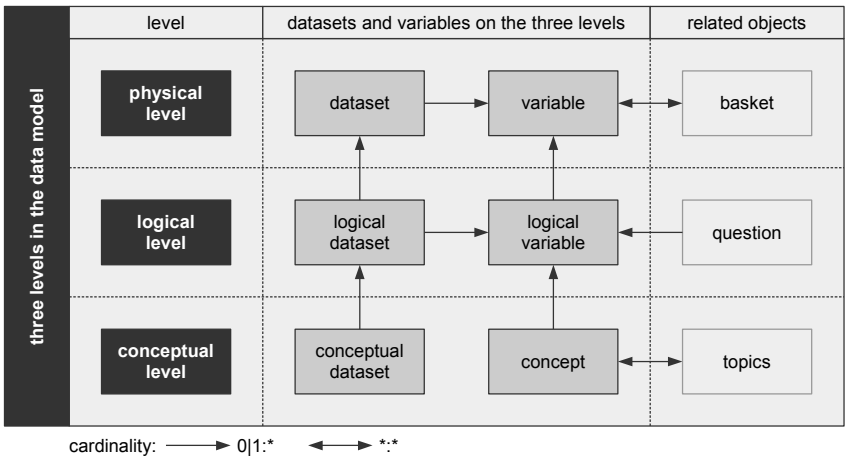

Figure 7.2: In DDI on Rails, variables are documented on three levels: The conceptual level links comparable variables over time and across studies. The logical level links multiple versions of one variables (if variables are distributed repeatedly). And the physical level represents the actual variable in a dataset.

queries to find variables and questions that are measuring the same concept in various studies.

DDI on Rails is designed to use three identifier systems: local identifiers (integers) that are generated by the web framework automatically, user-friendly names managed by researchers, and UUIDs for persistent identifiers. The web framework Ruby on Rails assigns, by default, an *integer* ID to each object, which is used as the primary key in the relational database and is independent from object attributes to enabling object-relational-mapping. These identifiers are specific to each instance of the software (starting with 1 in each instance), and are therefore no suitable candidates for persistent identifiers—a new installation would otherwise break the persistent identifiers. *Names* are alpha-numeric identifiers for various digital objects that are unique within a limited scope. Two datasets, for example, can both have a variable named `age`—this variable name is unique within each dataset but not among them. Names are also the way researchers usually refer to digital objects, because they are convenient to use. In DDI on Rails most imports and exports are based on these name-based identifiers. Names, however, can change over time and they are also not suitable candidates for persistent identifiers. Finally, an *Universal Unique Identifiers* (UUID) can be assigned to objects if reasonable. Not every object,

however, needs a persistent identifier. A variable, for example, should have one but not each category of a variable. Polymorphic associations, a specific feature of Ruby on Rails, allows a sophisticated implementation of those UUIDs that already has the character of an inverted index allowing the implementation of a efficient resolver. However, the final design of the persistent identifiers and, in particular, which registration agency to use, is still undecided. The most recent decision to run DDI on Rails as a hosted service on paneldata.org (see section 7.5) could define a reference.

The data model is designed with a focus on the relational database as part of the web application, considering the object-relational-mapping (ORM) in Ruby on Rails. As one example, Ruby on Rails promotes single table inheritance, where class hierarchies (e. g., questionnaires as a specialization of an instrument) are stored in a single table. The database design influenced the definition of a set of CSV files that are used for imports and exports. DDI on Rails supports common relational databases, including PostgreSQL [222], SQLite [240], and MySQL [246]. Constraints and validations are managed and processed on the application layer, while the database itself is used as a pure storage system. Thus meaning that no database-specific functionality (like the use of stored procedures in PostgreSQL) is used. The import files (CSV) basically have the same structure as the relational database. The implementation, however, differs in some key aspects—it uses names as identifiers and includes additional columns to make it more convenient to edit. The CSV formats are documented in detail on the DDI on Rails website [40].

7.3 User functionality

Starting with the homepage (figure A.1 on page 147), the user has various options to interact with the system. Most options are available directly, but some more personalized features are only available to registered users after login. The results from the user survey (section 1.5) suggest that researchers are, most of all, interested in finding and understanding variables which they can use for their publications. Therefore, the following description of the user functionality focuses on the variable documentation part of the system.

DDI on Rails offers three ways to find variables: the study browser, the search interface, and the topic list. The *study browser* bundles functionality to explore a particular study. The starting page includes the study description as introduced in section 3.1. Furthermore, the study browser includes browse and search function-

ality for datasets, variables, and questionnaires. An important part of the dataset browser is the documentation of multiple versions of the data, bundled as data distributions. The *search interface* (figure A.2 on page 148) covers concepts, datasets, variables, questionnaires, questions, and publications. It combines both full-text search and faceting. Faceting allows for the narrowing down of search results along of pre-defined categories. Variables, for example, can be selected by study, year, analysis unit, or type of variable. Furthermore, concepts are grouped in a *topic list* that works like the table of contents in a book. They are linked to concepts that group variables and questions. The topics are basically concept groups as introduced in section 5.4.

Variables are described by a basic set of metadata generated in r2ddi, which includes labels and basic statistics. Additionally, variables are linked to other variables and related materials using the mechanisms described in chapter 5 and section 7.2: associating variables with concepts, linking variables directly as data transformations, and utilizing the logical level for versioning. Based on these information, the variable-view links to concepts, underlying questions (if applicable), and related variables over time. As an additional feature for panel researchers, categorical variables over time are compared based on their category values and labels to indicate changes. Similar, questions are compared using a diff tool [247] to highlight even small changes. Figures A.3 to A.6 in the appendix illustrate these features.

After logging into the system, a workspace enables users to collect variables in a basket (figure A.7 on page 153), which is designed similar to shopping baskets on common e-commerce sites. For some studies, a script generator (figure A.8 on page 154) is available to create the respective Stata, SPSS, or R code to select the variables in the basket from the original datasets. In the original SOEPinfo, the basket was implemented on the client side (using JavaScript). DDI on Rails stores baskets on the server, enabling the system to provide additional functionality through the system's API, some of which is still under development. When, for example, a new version (distribution) of a study is published, the system can check for potential inconsistencies for all variables in a particular basket. Similar, DDI on Rails can look for related variables from other studies, encouraging re-analysis and supporting statistical matching. By default, a basket is only visible to its owner, but researchers can make baskets public in order to document the data they analyzed for a particular publication. In future versions, this functionality might be extended to a comprehensive re-analysis archive, complemented with persistent identifiers, the

documentation of publications, and upload functionality for related material, thus solving the problem that panel providers cannot allow their users to re-publish the data (see section 1.6).

The workspace is also the part of the system where data managers can update the metadata of a study or export and backup content. The two most important tasks are importing new metadata and indexing the metadata for search. Data managers can initiate both processes (import and index) from the web interface. Earlier versions (during development) enabled data managers to edit content directly online, but this functionality broke with the Single Source of Truth principle and was, consequently, removed.

7.4 Interoperability

Interoperability concerns two aspects of DDI on Rails: importing data into the system and making those data available to other applications (API, exports)—both map to the internal data model introduced in section 7.2. The formats for inputs and outputs, however, differ significantly because the use cases and requirements are different. Imports are prepared by data managers working for the studies that are documented. The imports are optimized for the data managers to be easy to edit and interoperable with the tools they use, which explains the focus on rectangular data formats. The exports, on the contrary, are optimized to be machine actionable, meaning, that they should be accessible and executable for other software products with a minimum of human interaction.

Study-specific *imports* are divided into five groups, which range from the most basic information about a study and its datasets to very detailed and comprehensive metadata about instruments and other related material. DDI on Rails is designed as a data portal and, therefore, information about variables is considered to have higher priority then information about instruments, for example. Table 7.1 gives an overview of all imports available to data managers. The core of the documentation is the study description, which is stored in Markdown format. The next level are the metadata about the datasets, which are automatically extracted from research datasets using r2ddi. Next, the description of datasets is complemented with additional information on the logical and conceptual level. Today, most studies do not have a semantic model of their questionnaires by default. While a comprehensive documentation of the instruments is desirable, data providers should not wait to document their data until they have a complete representation of all in-

Table 7.1: Imports for DDI on Rails. The first five imports are study-specific, ordered by their relevance. Studies should start with a basic study description and incrementally add further imports. The last three imports are study-independent.

	Import	Content
1	study description	Prose text description of the study, stored in Markdown.
2	dataset description	Endogenous and exogenous metadata about the data, extracted by r2ddi and provided as DDI-C compliant XML.
3	variable linkage	Description of the data on the logical and conceptual level, and data transformations in the respective CSV formats.
4	instruments	The semantic model of the questionnaires, provided as CSV files, queXML, or QeDML, complemented with images or PDFs of the field layout.
5	additional material	Files, logos, etc.
a	topics	Hierarchical representation of topics, stored in CSV.
b	publications	Publications in Endnote or BibTeX format.
c	classifications	Classifications for the search interface, describing time periods, conceptual datasets, or analysis units.

struments. Finally, other material and additional metadata can be added to provide more details. The study-specific imports are complemented with three groups of imports that affect all studies: the definition of concepts and topics, publications that are related to the data, and classifications to be used in the facets of the search interface.

Imports are stored in Git repositories (either on Github [215] or on Gitlab [214]) using a directory layout that is compliant with the previous one for data transformations (section 6.3). The imports are stored in a directory called `import/`, which contains the study description and all CSV files. The XML files for the dataset descriptions are stored in a sub-directory `r2ddi/`, which is further subdivided to identify the distributions and the language of the metadata. Further directories contain the semantic models for the instruments and other files. Data managers state the URL for the repositories API in the workspace of DDI on Rails and can afterwards start the import.

Most of the import formats are also available as *exports*. However, data managers are advised not to misinterpret DDI on Rails as a storage for metadata. In particular, the exports differ from the imports not only regarding the structure but also regarding the content as some information are stored differently than in the original import. The exports are, however, indispensable in facilitating the reuse of metadata across studies.

The exports are only one part of a comprehensive Application Programming Interface (API) that enables other programs to access the data (metadata) stored in DDI on Rails. In general, a RESTful interface structures the requests. Besides the human-readable HTML interface, most metadata are also available in JSON format. Furthermore, the code generated by the script generators is available via API, using a security token to protect the user content. Stata users, for example, can execute the code generated in DDI on Rails directly from Stata using the `do` command. This allows them to work with DDI on Rails interactively as changes on the basket are instantly available in Stata.

7.5 paneldata.org

The development of DDI on Rails was initiated to replace SOEPinfo [51]. The name 'DDI on Rails' denotes the software system, the actual implementation was originally intended to be called 'SOEPinfo v.2'. However, we decided to open the service for external studies. In this context, the name SOEPinfo v.2 seemed too narrow and the hosted service was renamed to 'paneldata.org' [41]. In addition to the SOEP-Core study and all SOEP-related studies, as of June 2015, two external studies use paneldata.org: the German Family Panel (Pairfam) and the German Internet Panel (GIP). This section takes a closer look how all these studies adopted their documentation to DDI on Rails. At the same time, unmet requirements are identified, which will be considered in the further development of DDI on Rails.

The SOEP was already introduced in detail in section 1.3. Two aspects of the SOEP-Core study are of particular interest for this assessment of DDI on Rails: the introduction of SOEPlong and the mobility of samples across studies. *SOEP-long* is a new version of the SOEP-Core data, which provides the data in the long format. When documenting these new data, the question arose whether to document SOEPlong as part of the SOEP-Core study or as a new study. In the final design, SOEPlong is documented as an independent study, which is basically a transformation of the SOEP-Core data. This adds a new use case to the list of vari-

able links in section 5.2 because the documentation of harmonized data in SOEP-long links back to the cross-sectional data in SOEP-Core. Another aspect of SOEP-Core is the documentation of *samples* and their mobility. The development of a sample-documentation in DDI on Rails has been postponed because the requirements started to change during development. The traditional design of samples covers the case where one study has multiple samples, but one sample always belongs to exactly one study. However, between 2011 and 2015, the SOEP started two projects where samples are exchanged between studies. Two samples from the original SOEP-Core became part of SOEP-IS, and the SOEP-related study Families in Germany are integrated in SOEP-Core.

The SOEP Innovation Sample (SOEP-IS) [70] and the Berlin Aging Study (BASE-II) [71, 248] represent the class of SOEP-related studies. In the terms of chapter 5, they also cover the use case where multiple studies are designed to be interoperable from the beginning. SOEP-IS and BASE-II reuse a significant number of instruments from SOEP-Core and they use the same identifiers for concepts. At least for the group of SOEP-related studies, the common set of concepts facilitates statistical matching to combine multiple data sources. The common set of concepts, however, presents the SOEP team with a new challenge, for example, to find new ways to manage the growing number of concepts across studies and with external partners (e. g., the fieldwork organization TNS Infratest).

In contrast to SOEP-related studies, the German Family Panel (Pairfam) [78, 88] and the German Internet Panel (GIP) [80] have been designed independently with different survey modes, other software tools, and no definition of concepts. Fortunately, both studies developed systems for consistent variable names that can be used to extract time-independent identifiers. However, the link to the concepts in paneldata.org, which are based on the SOEP, is still pending—while many other tasks can be more or less automated, matching variables to the respective concepts depends on human interpretation.

7.6 Further development

DDI on Rails provides a generic tool for data documentation and dissemination, specialized on panel data. The software is based on the web-framework Ruby on Rails and the DDI standard for metadata. However, it was not possible to map the XML-based DDI standard directly to the relational database that is used as the primary storage in DDI on Rails. Instead, the standard was used to design a rela-

tional database schema, which is implemented in the database and the CSV-based exchange formats. The core of the user functionality are the search interface, the study browser, and the workspace that supports researchers in assembling their personal use files. On the technical side, the system provides a set of CSV exports and a RESTful API for external software tools to access the metadata stored in DDI on Rails. DDI on Rails is used in production on paneldata.org [41] where both SOEP-related and external panel studies are documented.

Further plans for the development of DDI on Rails can be split in three groups: increasing the usability, extending the set of standardized metadata covered by the data model, and expanding the interoperability with external tools. The feedback from researchers using DDI on Rails focuses on improving the search functionality—both on the level of keyword search and browsing (links between elements). Regarding the data model, there are various new needs: the SOEP requires a detailed representation of samples, the GIP uses more complex questionnaire designs including experiments, and SOEPlong demands for an extended documentation of concatenated transformation processes and more detailed statistics for variables. On the technical side, it is intended to make all metadata available as linked open data.

Conclusion

Part I provides a generic perspective on panel studies to facilitate the development of reusable data models and software tools. To accomplish this, the first chapter takes a very specific look at one particular use case—the German Socio-Economic Panel (SOEP)—its workflow and the perspective of the data users. Researchers, both producing and using the panel data, usually have no background in software development or database management. Their preferred tools are currently Stata, SPSS, and R; but historical comparison makes clear that preferences can rapidly change. Furthermore, the panel researchers now prefer the long format, which is also the preferable format from a data modeling perspective.

Chapters 2 and 3 identify generic process steps and digital objects that are common to panel studies. The design of the Generic Longitudinal Business Process Model (GLBPM) proves that a generic model for panel studies if feasible. However, implementations for the digital objects are more diverse, limiting the possibilities to abstract from specific implementation details. The key objects are the study description, questionnaires, research data, transformation scripts, and the documentation (including long-term preservation). Based on the DDI Codebook standard, a simple outline for a prose study description is proposed, which is more flexible than the original DDI structure. Regarding questionnaires, we saw that the perspective of data modelers significantly differs from the perspective of survey methodologists. The later focus on survey modes while the former care more about the behavior of the questionnaire (static versus dynamic) and the various aspects of the documentation (semantic model, field layout, and source code). The main challenge regarding research data is the representation of missing values, which significantly differs for Stata, SPSS, and R. Furthermore, the proprietary formats of these packages might be more convenient for the researchers, but they do not provide a suitable solution for long-term preservation (archivists prefer ASCII formats

like CSV files). Transformation scripts are the most problematic digital object in respect to a generic representation. The VTL suggests itself as a possible solution, but it could at best present a reproducible but not a replicable representation. Finally, five steps towards an interoperable documentation and representation of a study and its digital objects are proposed: prose documentation, structured formats, open and plain text formats, standardized formats, and linked data.

Chapter 4 examines metadata. Originally considered to be part of the documentation, metadata are now accompanying the whole process. In a metadata-driven design, they are created even before the objects or tasks they describe. This enables us to automate tasks using metadata as an input. To comply with the Single Source of Truth principle, we distinguished exogenous and endogenous metadata. Endogenous metadata are only views on information that are managed within a digital object. Furthermore, chapter 4 selects a set of standards for digital objects and metadata to support the development of interoperable data models and software tools. Two examples of metadata that can support metadata-driven workflows are presented in chapter 5, which takes a closer look at common use cases for finding related variables and combining research data based on statistical matching or record linkage. Solutions for both omnidirectional and unidirectional relations are proposed. Omnidirectional relations utilize concepts as a common link. This structure is very flexible and can be used to combine basically everything, not only variables. Unidirectional relations, on the other hand, are usually based on transformations and are therefore modeled as process steps.

Part II proposes a reference architecture for panel studies and introduces the design and implementation of the data portal DDI on Rails. The reference architecture consists of three parts. First, a set of file formats for the digital objects and their metadata is defined. Second, corresponding software tools are selected that enable us to implement a metadata-driven infrastructure. And third, a standard directory layout for transformation processes is proposed, which facilitates automation and better documentation of those tasks. The reference architecture is intended as a proof of concept for the previous, more theoretical discussion. At the same time, it provides a software environment for the development of DDI on Rails—a data dissemination and discovery tool for panel studies. The functionality is focused on exactly one process step in the GLBPM ('data dissemination and discovery') to increase the re-usability of the software. The main features are the search interface (including references to related material based on the data structures proposed in

chapter 5) and a workspace that enables researchers to compose their individual data files. DDI on Rails is already used in production on paneldata.org.

The production of research data, in particular in panel studies, is mostly organized in silos. The design of a metadata-driven infrastructure, the focus of standardized formats and tools, as well as the claim for linked open data in combination make valuable first steps toward an open and interoperable infrastructure for research data. However, further work is required, both on the organizational and the technical level. Research organizations as well as individual researchers must adopt to standards for open data and publish their work. Initiatives like the DDI community or the Research Data Alliance (RDA) can support researcher, but ultimately it depends on the researchers and their organizations. On the technical level, linked open data provide a chance to link research data and, in particular, their metadata. However, it is still work in progress that existing infrastructures provide linked open data and that metadata standards adapt to the new possibilities.

The implementation of DDI on Rails and the documentation of existing panel studies on paneldata.org highlight the need for a joint infrastructure to curate concepts, which link the content of panel studies on the level of their variables and questions. Chapter 5 proposes the use of a concept registry. The content of this registry could be curated by participating studies, could be published as linked open data, and would facilitate new chances for record linkage and statistical matching. Furthermore, linking variables and questions through concepts is a crucial requirement to enable metadata-driven infrastructures (chapter 4). Thus, the development of such a registry would be a reasonable follow up project to this dissertation and would also complement the development of DDI on Rails.

List of Figures

List of Tables

References

[1] Mary Vardigan. The DDI matures: 1997 to the present. *IASSIST Quarterly*, 37(1):45–50, 2013.

[2] Jeremy Iverson. Metadata-driven survey design. *IASSIST Quarterly*, 33(1):7–9, 2009.

[3] Pedro Revilla, José Luis Maldonado, Francisco Hernández, and José Manuel Bercebal. Implementing a corporate-wide metadata driven production process at INE Spain. *Instituto Nacional de Estadística, Working Paper*, 5:2012, 2012.

[4] Bo Sundgren. Documentation and quality in official statistics. In *EU conference on Quality in Statistics*, Stockholm, Sweden, 2001.

[5] Susanne Rässler. Data fusion: identification problems, validity, and multiple imputation. *Austrian Journal of Statistics*, 33(1-2):153–171, 2004.

[6] Florian Meinfelder. Datenfusion: Theoretische Implikationen und praktische Umsetzung. In Thomas Riede, Sabine Bechtold, and Notburga Ott, editors, *Weiterentwicklung der amtlichen Haushaltsstatistiken*, pages 83–98. Scivero, Berlin, Germany, 2013.

[7] Andreas Schmitz, Jan Skopek, Florian Schulz, Doreen Klein, and Hans-Peter Blossfeld. Indicating mate preferences by mixing survey and process-generated data. the case of attitudes and behaviour in online mate search. *Historical Social Research / Historische Sozialforschung*, 34(1):77–93, 2009.

[8] Christin Czaplicki and Julie Korbmacher. SHARE-RV: Verknüpfung von Befragungsdaten des Survey of Health, Ageing and Retirement in Europe mit administrativen Daten der Rentenversicherung. *Gesundheit, Migration und Einkommensungleichheit*, pages 28–37, 2010.

[9] Carol Tenopir, Suzie Allard, Kimberly Douglass, Arsev Umur Aydinoglu, Lei Wu, Eleanor Read, Maribeth Manoff, and Mike Frame. Data sharing by scientists: practices and perceptions. *PLoS ONE*, 6(6):e21101, 2011.

[10] Code share: Papers in Nature journals should make computer code accessible where possible. *Nature*, 514:536, 2014.

[11] Liz Allen, Amy Brand, Jo Scott, Micah Altman, and Marjorie Hlava. Credit where credit is due. *Nature*, 508:312–313, 2014.

[12] Jonathan Schooler. Metascience could rescue the 'replication crisis'. *Nature*, 515:9, 2014.

[13] Greg Wilson, D.A. Aruliah, C. Titus Brown, Neil P. Chue Hong, Matt Davis, Richard T. Guy, Steven H.D. Haddock, Kathryn D. Huff, Ian M. Mitchell, Mark D. Plumbley, et al. Best practices for scientific computing. *PLoS biology*, 12(1):e1001745, 2014.

[14] Jeffrey M. Perkel. Pick up Python. *Nature*, 518:125–126, 2015.

[15] Sylvia Tippmann. Programming tools: Adventures with R. *Nature*, 517:109–110, 2015.

[16] Bruce E. Bargmeyer and Daniel W. Gillman. Metadata standards and metadata registries: An overview. In *Proceedings of the Second International Conference on Establishment Surveys: Survey Methods for Businesses, Farms, and Institutions Buffalo. Citeseer*, Buffalo, USA, 2000. ICES, Citeseer,.

[17] Erik Duval. Metadata standards: What, who & why. *Journal of Universal Computer Science*, 7(7):591–601, 2001.

[18] Arofan Gregory, Pascal Heus, and Jostein Ryssevik. Metadata. RatSWD Working Paper Series 57, German Council for Social and Economic Data (RatSWD), Berlin, Germany, 2009.

[19] Wendy Thomas, Arofan Gregory, J. Gager, Jon Johnson, and Joachim Wackerow. Technical specification, part I: Technical documentation. version 3.2. Standard, DDI Allliance, 2014.

[20] Wendy Thomas, Arofan Gregory, J. Gager, and Jon Johnson. Technical specification, part II: User guide. version 3.2. Standard, DDI Alliance, 2014.

[21] Grant Blank and Karsten Boye Rasmussen. The Data Documentation Initiative the value and significance of a worldwide standard. *Social Science Computer Review*, 22(3):307–318, 2004.

[22] Karsten Boye Rasmussen. Social science metadata and the foundations of the DDI. *IASSIST Quarterly*, 37(1):28–35, 2013.

[23] John Haisken-DeNew and Joachim R. Frick. DTC: Desktop Companion to the German Socio-Economic Panel (SOEP). *Deutsches Institut für Wirtschaftsforschung, Berlin*, 2005.

[24] Simonetta Longhi and Alita Nandi. *A Practical Guide to Using Panel Data*. SAGE, 2014.

[25] Jürgen Schupp. Paneldaten für die Sozialforschung. In *Handbuch Methoden der empirischen Sozialforschung*, pages 925–939. Springer, 2014.

[26] Armando Fox and David Patterson. *Engineering Software as a Service: An Agile Approach Using Cloud Computing*. Strawberry Canyon LLC, Kindle edition, 2014.

[27] Avraham Leff and James T. Rayfield. Web-application development using the model/view/controller design pattern. In *Enterprise Distributed Object Computing Conference, 2001. EDOC'01. Proceedings. Fifth IEEE International*, pages 118–127. IEEE, 2001.

[28] Jack Greenfield and Keith Short. Software factories: assembling applications with patterns, models, frameworks and tools. In *Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, pages 16–27. ACM, 2003.

[29] Tim O'Reilly. What is Web 2.0: Design patterns and business models for the next generation of software. *Communications and Strategies*, 65(1):17–37, 2007.

[30] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data – the story so far. *International journal on semantic web and information systems*, 5(3):1–22, 2009.

[31] Tim Berners-Lee. Linked data. `http://www.w3.org/DesignIssues/LinkedData.html`, last accessed 2015-01-26.

[32] L. Fernando Ramos Simón, Rosario Arquero Avilés, Iuliana Botezan, Félix del Valle Gastaminza, and Silvia Cobo Serrano. Open data as universal service. new perspectives in the information profession. *Procedia-Social and Behavioral Sciences*, 147:126–132, 2014.

[33] William Block, Thomas Bosch, Bryan Fitzpatrick, Dan Gillman, Jay Greenfield, Arofan Gregory, Marcel Hebing, Larry Hoyle, Chuck Humphrey, Jon Johnson, Jenny Linnerud, Brigitte Mathiak, Steven McEachern, Olof Olsson, Barry Radler, Ornulf Risnes, Dan Smith, Wendy Thomas, Joachim Wackerow, Dennis Wegener, and Wolfgang Zenk-Möltgen. Developing a model-driven DDI specification. `http://www.ddialliance.org/system/files/DevelopingaModel-DrivenDDISpecification2013_05_15.pdf`, last accessed 2015-01-14, 2012.

[34] Benedikt Fecher, Sascha Friesike, and Marcel Hebing. What drives academic data sharing? *PLoS ONE*, 10(2), 2015.

[35] Benedikt Fecher, Sascha Friesike, Marcel Hebing, Stephanie Linek, and Armin Sauermann. A reputation economy: Results from an empirical survey on academic data sharing. Discussion Paper 1454, DIW Berlin, Berlin, Germany, 2015.

[36] Andreas Schmitz, Olga Yanenko, and Marcel Hebing. Identifying artificial actors in e-dating: A probabilistic segmentation based on interactional pattern analysis. In *Challenges at the Interface of Data Analysis, Computer Science, and Optimization*, pages 319–327. Springer, 2012.

[37] Ingo Barkow, William Block, Jay Greenfield, Arofan Gregory, Marcel Hebing, Larry Hoyle, and Wolfgang Zenk-Möltgen. Generic Longitudinal Business Process Model. DDI Working Paper Series 5, Data Documentation Initiative, 2013.

[38] Marcel Hebing, Florian Griese, Janine Napieraj, Marius Pahl, Carolin Stolpe, and Gert G. Wagner. Zur Struktur von empirischen Sozial-, Verhaltens- und Wirtschaftsforschern – Ein Überblick über die Ergebnisse der SOEP-Nutzerbefragungen. SOEPpapers on Multidisciplinary Panel Data Research 708, DIW-SOEP, Berlin, Germany, 2014.

[39] Marcel Hebing and Jan Goebel. r2ddi. Software, `https://github.com/ddionrails/r2ddi`.

[40] ddionrails.org. `http://www.ddionrails.org`, last accessed 2015-03-27.

[41] paneldata.org. Software. `https://paneldata.org`, last accessed 2015-03-27.

[42] Veerle Van den Eynden, Louise Corti, Matthew Woollard, Libby Bishop, and Laurence Horton. *Managing and Sharing Data – Best Practices for Researchers*. UK Data Archive, 2011.

[43] Gert G. Wagner, Joachim R. Frick, and Jürgen Schupp. The German Socio-Economic Panel Study (SOEP) – scope, evolution and enhancements. *Schmollers Jahrbuch*, 127(1):139–169, 2007.

[44] Heather Laurie. Panel studies. Oxford Bibliographies: `http://www.oxfordbibliographies.com/view/document/obo-9780199756384/obo-9780199756384-0108.xml`, last accessed 2015-01-20.

[45] Hans Rattinger, Sigrid Roßteutscher, Rüdiger Schmitt-Beck, Harald Schoen, Bernhard Weßels, and Christof Wolf. German Longitudinal Election Study (GLES). Study. `http://gles.eu/wordpress/english/design/`, last accessed 2015-04-15.

[46] Martin Kroh. Documentation of sample sizes and panel attrition in the German Socio Economic Panel (SOEP) (1984 until 2012). Data documentation, DIW Berlin, Berlin, Germany, 2013.

[47] Sonderforschungsbereich 3. Antrag auf Förderung des Projekts B-5: Das Sozio-ökonomische Panel, 1982.

[48] Ingo Sieber. SOEP samples overview - 2012 / wave 29. `http://panel.gsoep.de/soepinfo2012/info/soep_samples_size.pdf`, last accessed 2015-03-27, 2013.

[49] SIR database software. Software. http://www.sir.com.au/, last accessed 2015-05-09.

[50] Andreas Bauer and Holger Günzel. *Data-Warehouse-Systeme: Architektur, Entwicklung, Anwendung*. dpunkt. verlag, 2013.

[51] Ingo Sieber et al. SOEPinfo. Software. http://panel.gsoep.de/soepinfo/, last accessed 2015-03-27.

[52] German Socio-Economic Panel (SOEP). Study. http://diw.de/en/soep, last accessed 2015-05-14.

[53] Joachim R. Frick, Jan Goebel, Michaela Engelmann, Uta Rahmann, et al. The research data center (RDC) of the German Socio-Economic Panel (SOEP). *Schmollers Jahrbuch*, 130(3):393, 2010.

[54] Deborah A. Bowen, Michaela Engelmann, Sabine Kallwitz, Christine Kurka, and Uta Rahmann. Entwicklung des SOEPservice. *Vierteljahrshefte zur Wirtschaftsforschung*, 77(3):130–141, 2008.

[55] Sandra Gerstorf and Jürgen Schupp. Soep wave report 2014. Technical report, German Socio-Economic Panel (SOEP), 2015.

[56] John P. Haisken-DeNew and Markus Hahn. Panelwhiz: A flexible modularized Stata interface for accessing large scale panel data sets. http://www.panelwhiz.eu/docs/PanelWhiz_Introduction.pdf, last accessed 2015-01-26, 2006.

[57] Ute Hanefeld. *Das Sozio-ökonomische Panel – Grundlagen und Konzeption*. Campus, Frankfurt/New York, 1987.

[58] Marcel Hebing. DDI on Rails. Presentation at the 11th International German Socio-Economic Panel User Conference, Berlin, 2014.

[59] Carsten Schmitz et al. Limesurvey: An open source survey tool. Software. http://www.limesurvey.org, last accessed 2015-04-15.

[60] William W. Gould and Nicholas J. Cox. History of Stata. http://www.stata.com/support/faqs/resources/history-of-stata/, last accessed 2015-03-01, 2014.

[61] John M. Chambers. Users, programmers, and statistical software. *Journal of Computational and Graphical Statistics*, 9(3):404–422, 2000.

[62] Roger D. Peng. Reproducible research in computational science. *Science*, 334(6060):1226–1227, 2011.

[63] Gary King. Replication, replication. *PS: Political Science & Politics*, 28(03):444–452, 1995.

[64] Daniel Kleppner and Phillip A. Sharp. Research data in the digital age. *Science*, 325(5939):368–368, 2009.

[65] German Socio-Economic Panel (SOEP). Archive for re-analysis of published findings. `http://www.diw.de/de/diw_01.c.340858.de/soep_re_analyses.html`, last accessed 2015-05-09.

[66] Peter Lynn. A quality framework for longitudinal studies. `https://www.iser.essex.ac.uk/files/ulsc/standards/framework/2001-09-26.pdf`, last accessed 2015-01-17, 2001.

[67] Data Documentation Initiative (DDI). `http://www.ddialliance.org`, last accessed 2015-04-07.

[68] Mary Vardigan. DDI timeline. *IASSIST Quarterly*, 37(1):51–56, 2013.

[69] Ann Green and Chuck Humphrey. Building the DDI. *IASSIST Quarterly*, 37(1):36–44, 2013.

[70] David Richter and Jürgen Schupp. SOEP Innovation Sample (SOEP-IS): Description, structure and documentation. SOEPpapers on Multidisciplinary Panel Data Research 463, DIW-SOEP, Berlin, Germany, 2012.

[71] Anke Böckenhoff, Denise Sassenroth, Martin Kroh, Thomas Siedler, Peter Eibich, and Gert G. Wagner. The socio-economic module of the Berlin Aging Study II (SOEP-BASE): Description, structure, and questionnaire. SOEPpapers on Multidisciplinary Panel Data Research 568, DIW-SOEP, Berlin, Germany, 2013.

[72] TNS Infratest Sozialforschung. SOEP-RS BASE II 2008–2012: Erhebungsinstrumente der Berliner Altersstudie II. SOEP Survey Papers 137, DIW-SOEP, Berlin, Germany, 2013.

[73] Mathis Schröder, Rainer Siegers, and C. Katharina Spieß. Familien in Deutschland – FiD. *Schmollers Jahrbuch*, 133:595–606, 2013.

[74] „Familien in Deutschland (FiD)“ in Kürze. `http://diw.de/documents/dokumentenarchiv/17/diw_01.c.368261.de/fid_in_k%C3%BCrze.pdf`, last accessed 2014-01-19.

[75] Gesis, NEPS, and SOEP. PIAAC Panel (PIAAC-L). Study. `http://www.gesis.org/en/research/external-funding-projects/projektuebersicht-drittmittel/piaac-panel-piaac-l/`.

[76] Martin Diewald, Rainer Riemann, Frank M. Spinath, et al. TwinLife. Study. `http://www.twin-life.de/en`.

[77] Hans-Peter Blossfeld, Hans-Günther Roßbach, and Jutta von Maurice, editors. *Education as a Lifelong Process*. VS Verlag für Sozialwissenschaften, 2011.

[78] Johannes Huinink, Josef Brüderl, Bernhard Nauck, Sabine Walper, Laura Castiglioni, and Michael Feldhaus. Panel analysis of intimate relationships and family dynamics (pairfam): Conceptual framework and design. *Zeitschrift für Familienforschung*, 23(1):77–101, 2011.

[79] Axel Börsch-Supan, Martina Brandt, Christian Hunkler, Thorsten Kneip, Julie Korbmacher, Frederic Malter, Barbara Schaan, Stephanie Stuck, and Sabrina Zuber. Data resource profile: the Survey of Health, Ageing and Retirement in Europe (SHARE). *International Journal of Epidemiology*, 42:992–1001, 2013.

[80] SFB 884. German Internet Panel (GIP). Study. `http://reforms.uni-mannheim.de/internet_panel/home/`.

[81] Gundi Knies. Understanding Society: The UK household longitudinal study, waves 1-4 (version 1.1). User manual, Institute for Social and Economic Research, University of Essex, Colchester, UK, 2014.

[82] Michelle Summerfield, Simon Freidin, Markus Hahn, Ning Li, Ninette Macalalad, Laura Mundy, Nicole Watson, Roger Wilkins, and Mark Wooden. HILDA user manual – release 13. User manual, Melbourne Institute of Applied Economic and Social Research, University of Melbourne, Melbourne, Australia, 2014.

[83] Korea Employment Information Service. Korean Labor & Income Panel Study (KLIPS). Study. `http://eng.keis.or.kr/eng/project/survey/laborer.jsp`.

[84] Panel Study of Income Dynamics (PSID). PSID main interview user manual: Release 2013. User manual, Institute for Social Research, University of Michigan, Ann Arbor, USA, 2013.

[85] Higher School of Economics (HSE). Russia Longitudinal Monitoring Survey (RLMS). Study. `http://www.cpc.unc.edu/projects/rlms-hse`.

[86] Marieke Voorpostel, Robin Tillmann, Florence Lebert, Ursina Kuhn, Oliver Lipps, Valérie-Anne Ryser, Flurina Schmid, Erika Antal, and Boris Wernli. Swiss household panel user guide (1999 - 2013). User Guide Wave 15, FORS, Lausanne, Switzerland, 2014.

[87] Statistics Canada. Survey of Labour and Income Dynamics (SLID). Study. `http://www23.statcan.gc.ca/imdb/p2SV.pl?Function=getSurvey&SDDS=3889`.

[88] Oliver Arránz Becker, Josef Brüderl, Petra Buhr, Laura Castiglioni, Daniel Fuß, Volker Ludwig, Claudia Schmiedeberg, Jette Schröder, and Nina Schumann. The German family panel: Study design and cumulated field report (waves 1 to 5). Pairfam Technical Paper 1, Pairfam, Germany, 2014.

[89] Richard V. Burkhauser, Barbara A. Butrica, Mary C. Daly, and Dean R. Lillard. The Cross-National Equivalent File: A product of cross-national research. In Irene Becker, Notburga Ott, and Gabriele Rolf, editors, *Soziale Sicherung in einer dynamischen Gesellschaft (Social Insurance in a Dynamic Society)*. Campus, 2001.

[90] Joachim R. Frick, Stephen P. Jenkins, Dean R. Lillard, Oliver Lipps, and Mark Wooden. The Cross-National Equivalent File (CNEF) and its member country household panel studies. *Schmollers Jahrbuch*, 127(4):627–654, 2007.

[91] Dean R. Lillard, Rebekka Christopoulou, Jan Goebel, Simon Freidin, Ahmed Jaber, Oliver Lipps, Jim Vajionis, and KLIPS Team. The Cross-National Equivalent Files 1970–2009. `http://cnef.ehe.osu.edu/files/2012/11/CNEF_codebooks_All.pdf`, last accessed 20 Jan 2015, 2012.

[92] William C. Block, Christian Bilde Andersen, Daniel E. Bontempo, Arofan Gregory, Stan Howald, Douglas Kieweg, and Barry T. Radler. Documenting a wider variety of data using the data documentation initiative 3.1. DDI Working Paper Series 1, Data Documentation Initative, 2011.

[93] Sue Ellen Hansen, Jeremy Iverson, Uwe Jansen, Hilde Orten, and Johanna Vompras. Enabling longitudinal data comparison using DDI. DDI Working Paper Series – Longitudinal Best Practice 2, Data Documentation Initiative, 2011.

[94] Larry Hoyle, Fortunato Castillo, Benjamin Clark, Neeraj Kashyap, Denise Perpich, Joachim Wackerow, and Knut Wenzig. Metadata for the longitudinal data lifecycle. DDI Working Paper Series 3, Data Documentation Initiative, 2011.

[95] Stefan Kramer, Randy Banks, Vicky Chang, Ingo Sieber, Mary Vardigan, and Wolfgang Zenk-Möltgen. Presenting longitudinal studies to end users effectively using DDI metadata. DDI Working Paper Series – Longitudinal Best Practice 4, Data Documentation Initiative, 2011.

[96] United Nations Economic Commission for Europe (UNECE). Generic Statistical Business Process Model GSBPM. version 5.0. Technical report, UNECE, December 2013.

[97] Blagica Novkovska, Helena Papazoska, and Biljana Ristevska-Karajovanovikj. The gsbpm contribution to statistical business process standardization. In *European conference on Quality in official statistics, Athens*, 2012.

[98] DDI Alliance. DDI Codebook 2.5. Standard. `http://www.ddialliance.org/Specification/DDI-Codebook/2.5/`, 2012.

[99] DDI Alliance. element <stdyDscr> (fieldlevel documentation for DDI Codebook, version 2.5). http://www.ddialliance.org/Specification/DDI-Codebook/2.5/XMLSchema/field_level_documentation_files/schemas/codebook_xsd/elements/stdyDscr.html, last accessed 2015-02-14.

[100] DDI Alliance. DDI Lite – for DDI Codebook version 2.0 (tree structure). http://www.ddialliance.org/sites/default/files/ddi-lite.html, last accessed 2015-01-24.

[101] Wayne C. Booth, Gregory G. Colomb, and Joseph M. William. *The Craft Of Research (Chicago Guides To Writing, Editing, And Publishing)*. University Of Chicago Press, Chicago, USA, 3 edition, 2008.

[102] William S. Aquilino. Interview mode effects in surveys of drug and alcohol use: A field experiment. *Public Opinion Quarterly*, 58(2):210–240, 1994.

[103] Samuel Spencer. A case against the skip statement. http://bit.ly/CaseAgainstSkip, last accessed 2014-09-22, 2012.

[104] NIPO Software. NIPO ODIN (version 5.17). Software http://www.niposoftware.com/OnlineDocumentation/NIPO%20ODIN%205.17%20Scripter%27s%20Guide/index.htm?toc.htm?7134.htm, last accessed 2015-05-11.

[105] Australian Consortium for Social and Political Research Incorporated. queXML. Standard, http://quexml.sourceforge.net/, last accessed 2015-02-21.

[106] Ulrich Kohler and Frauke Kreuter. *Data analysis using Stata*. Stata Press, 2005.

[107] Cynthia A. Brandt, Richard Morse, Keri Matthews, Kexin Sun, Aniruddha M. Deshpande, Rohit Gadagkar, Dorothy B. Cohen, Perry L. Miller, and Prakash M. Nadkarni. Metadata-driven creation of data marts from an eav-modeled clinical research database. *International journal of medical informatics*, 65(3):225–241, 2002.

[108] Valentin Dinu and Prakash Nadkarni. Guidelines for the effective use of entity–attribute–value modeling for biomedical databases. *International Journal of Medical Informatics*, 76(11):769–779, 2007.

[109] Donald B. Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

[110] StataCorp. *Stata User's Guide – Release 13*. Stata Press, College Station, USA, 2013.

[111] StataCorp. *Stata Data Management Reference Manual – Release 13*. Stata Press, College Station, USA, 2013.

[112] IBM. *IBM SPSS Data Preparation 22*. IBM Software Group, Chicago, USA.

[113] IBM. *IBM SPSS Missing Values 22*. IBM Software Group, Chicago, USA.

[114] R Core Team. R language definition (version 3.1.2). R Manuals, `http://cran.r-project.org/doc/manuals/r-release/R-lang.pdf`, last accessed 2015-01-26, 2014.

[115] W. N. Venables, D. M. Smith, and the R Core Team. An introduction to r (version 3.1.2). R Manuals, `http://cran.r-project.org/doc/manuals/r-release/R-intro.pdf`, last accessed 2015-01-26, 2014.

[116] Adrian Duşa. DDI with R (R package "DDIwR", version 0.2). Comprehensive R Archive Network (CRAN). `http://cran.r-project.org/web/packages/DDIwR/index.html`, last accessed 2015-04-15, 2014.

[117] Ctrl alt share. *Scientific Data*, 2(4), 2015.

[118] Linus Torvalds, Junio Hamano, et al. Git. Software. `http://git-scm.com/`, last accessed 2015-03-19.

[119] Scott Chacon. *Pro Git*. Apress, New York, USA, 2009.

[120] Georg Brandl et al. Sphinx – Python documentation generator. Software. `http://sphinx-doc.org/`, last accessed 2015-03-19.

[121] Dimitri van Heesch et al. Doxygen. Software. `http://doxygen.org/`.

[122] Foteini Andrikopoulou, Luigi Bellomarini, Marc Bouffard, Vincenzo Del Vecchio, Fabio Di Giovanni, Stratos Nikoloutsos, Michele Romanelli, Laura Vignola, and Nikolaos Zisimos. Validation & Transformation Language: General description (version 0.12, draft). SDMX Technical Working Group – VTL Task Force: `http://sdmx.org/wp-content/uploads/2014/09/VTL_draft012_201409_part1_general.pdf`, last accessed 2015-01-22, 2014.

[123] Foteini Andrikopoulou, Luigi Bellomarini, Marc Bouffard, Vincenzo Del Vecchio, Fabio Di Giovanni, Stratos Nikoloutsos, Michele Romanelli, Laura Vignola, and Nikolaos Zisimos. Validation & Transformation Language: Library of operators (version 0.12, draft). SDMX Technical Working Group – VTL Task Force: `http://sdmx.org/wp-content/uploads/2014/09/VTL_draft012_201409_part2_operators2.pdf`, last accessed 2015-01-22, 2014.

[124] Chris Drummond. Replicability is not reproducibility: nor is it good science. In *Proceedings of the Evaluation Methods for Machine Learning Workshop at the 26th ICML*, Montreal, Canada, 2009.

[125] Louise Corti, Veerle Van den Eynden, Libby Bishop, and Matthew Woollard. *Managing and sharing research data: A guide to good practice*. Sage, 2014.

[126] German Socio-Economic Panel Study (SOEP). Item correspondence list (1984–2012). `http://panel.gsoep.de/items/items2012-2013-11-18.xls`, last accessed 2015-01-17, 2013.

[127] Thomas Bosch, Franck Cotton, Richard Cyganiak, Arofan Gregory, Benedikt Kämpgen, Olof Olsson, Heiko Paulheim, Joachim Wackerow, and Benjamin Zapilko. Ddi-rdf discovery vocabulary: A vocabulary for publishing metadata about data sets (research and survey data) into the web of linked data (unofficial draft 16 may 2014). http://rdf-vocabulary.ddialliance.org/discovery.html, last accessed 2015-02-27, 2014.

[128] Wendy Thomas. Mapping out a region of interacting standards. In *Proc. 58th World Statistical Congress*, pages 4107–4111, Dublin, Ireland, 2011.

[129] Pascal Heus and Arofan Gregory. Maximizing the potential of data – modern IT tools, best practices, and metadata standards for SBE sciences. http://odaf.org/papers/201010_Heus_Pascal_268.pdf, last accessed 2015-01-24, 2010.

[130] Otto K. Ferstl and Elmar J. Sinz. *Grundlagen der Wirtschaftsinformatik, Bd.1*. Oldenbourg Wissenschaftsverlag, 2000.

[131] Markus Lanthaler. *Third Generation Web APIs – Bridging the Gap between REST and Linked Data*. PhD thesis, Graz University of Technology, Austria, 2014.

[132] Inter-university Consortium for Political and Social Research (ICPSR). *Guide to Social Science Data Preparation and Archiving: Best Practice Throughout the Data Life Cycle*. Institute for Social Research University of Michigan, Ann Arbor, USA, 5th edition, 2012.

[133] Open Knowledge Foundation Labs. Tabular Data Package (version 1.0-beta-2). http://dataprotocols.org/tabular-data-package/, last accessed 2015-01-22, 2014.

[134] Erik Duval, Wayne Hodgins, Stuart Sutton, and Stuart L. Weibel. Metadata principles and practicalities. *D-lib Magazine*, 8(4):16, 2002.

[135] Paul Edwards, Matthew S. Mayernik, Archer Batcheller, Geoffrey Bowker, and Christine Borgman. Science friction: Data, metadata, and collaboration. *Social Studies of Science*, 41(5):667–690, 2011.

[136] Reedy Feggins. Establishing a "single source of truth". IBM DevOps Best Practice, https://www.ibm.com/developerworks/community/blogs/c914709e-8097-4537-92ef-8982fc416138/entry/devops_best_practice_-_establishing_a_%E2%80%9Csingle_source_of_truth%E2%80%9D?lang=en, last accessed 2015-01-25, 2014.

[137] Edward J. Wegman and Faleh Alshameri. On the extraction of endogenous metadata for text and image databases. Electronic Proceedings of: Knowledge Extraction and Modelling, Anacapri, Italy, http://www.stat.unipg.it/iasc/Proceedings/2006/COMPSTAT_Satellites/KNEMO/Lavori/Papers%20CD/Wegman%20Alshameri.pdf last accessed 2015-01-22, 2006.

[138] Bill Kules and Ben Shneiderman. Designing a metadata-driven visual information browser for federal statistics. In *Proceedings of the 2003 annual national conference on Digital government research*, pages 1–6. Digital Government Society of North America, 2003.

[139] Chiquito Crasto, Luis Marenco, Perry Miller, and Gordon Shepherd. Olfactory receptor database: a metadata-driven automated population from sources of gene and protein sequences. *Nucleic acids research*, 30(1):354–360, 2002.

[140] Charles Crichton, Jim Davies, Jeremy Gibbons, Steve Harris, Andrew Tsui, and James Brenton. Metadata-driven software for clinical trials. In *Proceedings of the 2009 ICSE Workshop on Software Engineering in Health Care*. IEEE Computer Society, 2009.

[141] David S. Janzen and Hossein Saiedian. Test-driven development: Concepts, taxonomy, and future direction. *Computer Science and Software Engineering*, page 33, 2005.

[142] E. Michael Maximilien and Laurie Williams. Assessing test-driven development at ibm. In *Software Engineering, 2003. Proceedings. 25th International Conference on*, pages 564–569. IEEE, 2003.

[143] Boby George and Laurie Williams. A structured experiment of test-driven development. *Information and software Technology*, 46(5):337–342, 2004.

[144] Reid Kaufmann and David Janzen. Implications of test-driven development: a pilot study. In *Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, pages 298–299. ACM, 2003.

[145] Laurie Williams, E. Michael Maximilien, and Mladen Vouk. Test-driven development as a defect-reduction practice. In *Software Reliability Engineering, 2003. ISSRE 2003. 14th International Symposium on*, pages 34–45. IEEE, 2003.

[146] DDI Alliance. Field level documentation. version 3.2. `http://www.ddialliance.org/Specification/DDI-Lifecycle/3.2/XMLSchema/FieldLevelDocumentation/`, last accessed 2014-01-17, 2014.

[147] ISO 26324. Information and documentation – digital object identifier system. ISO/IEC 11179-1:2004(E), International Organization for Standardization, Geneva, Switzerland, 2012.

[148] Dublin Core Metadata Initiative. Dublin Core Metadata Element Set, version 1.1. `http://dublincore.org/documents/2012/06/14/dces/`, last accessed 2015-01-23, 2012.

[149] United Nations Economic Commission for Europe (UNECE). Generic Statistical Information Model (GSIM): Specification (version 1.1). http://www1.unece.org/stat/platform/download/attachments/97356610/GSIM%20Specification%201__1.pdf?version=3&modificationDate=1388474373573&api=v2, last accessed 2015-01-24, 2013.

[150] ISO 11179. Information technology – metadata registries (MDR) – part 1: Framework. ISO/IEC 11179-1:2004(E), International Organization for Standardization, Geneva, Switzerland, 2004.

[151] ISO 11179. Information technology – metadata registries (MDR) – part 2: Classification. ISO/IEC 11179-2:2005(E), International Organization for Standardization, Geneva, Switzerland, 2004.

[152] ISO 11179. Information technology – metadata registries (MDR) – part 4: Formulation of data definitions. ISO/IEC 11179-4:2004(E), International Organization for Standardization, Geneva, Switzerland, 2004.

[153] ISO 11179. Information technology – metadata registries (MDR) – part 5: Naming and identification principles. ISO/IEC 11179-5:2005(E), International Organization for Standardization, Geneva, Switzerland, 2004.

[154] ISO 11179. Information technology – metadata registries (MDR) – part 6: Registration. ISO/IEC 11179-6:2005(E), International Organization for Standardization, Geneva, Switzerland, 2004.

[155] ISO 11179. Information technology – metadata registries (MDR) – part 3: Registry metamodel and basic attributes. ISO/IEC 11179-3:2013(E), International Organization for Standardization, Geneva, Switzerland, 2003.

[156] Consultative Committee for Space Data Systems (CCSDS). Reference model for an Open Archival Information System (OAIS). http://public.ccsds.org/publications/archive/650x0m2.pdf, last accessed 2015-01-24, 2012.

[157] BIS (Bank for International Settlements), ECB (European Central Bank), EUROSTAT (Statistical Office of the European Union), IMF (International Monetary Fund), OECD (Organization for Economic Co-operation and Development), UN (United Nations), and World Bank. SDMX 2.1 technical specification – consolidated version 2013. Standard. http://sdmx.org.

[158] Arofan Gregory and Pascal Heus. DDI and SDMX: Complementary, not competing, standards. Open Data Foundation, 2007.

[159] Rufus Pollock, Matthew Brett, and Martin Keegan. Data Packages (version 1.0-beta.10). http://dataprotocols.org/data-packages/, last accessed 2015-01-22, 2014.

[160] Open Knowledge Foundation Labs. JSON Table Schema (version 1.0-pre4). http://dataprotocols.org/json-table-schema/, last accessed 2015-01-22, 2013.

[161] Open Knowledge Foundation Labs. CSV Dialect Description Format (CSVDDF, version 1.2). http://dataprotocols.org/csv-dialect/, last accessed 2015-01-22, 2014.

[162] ISO 15836. Information and documentation – the Dublin Core metadata element set. ISO/IEC 15836, International Organization for Standardization, Geneva, Switzerland, 2009.

[163] Alerk Amin, Ingo Barkow, Stefan Kramer, David Schiller, and Jeremy Williams. Representing and utilizing DDI in relational databases. RatSWD Working Paper Series 191, German Council for Social and Economic Data (RatSWD), Berlin, Germany, 2012.

[164] Philology. Questionnaire editing & deployment markup language. Standard, http://www.philology.me/qedml, last accessed 2015-02-21.

[165] Philipp Lenssen et al. QuestML (QML). Standard, http://questml.com/, last accessed 2015-02-21.

[166] Victoria McNeil. How to convert a survey from Word to queXML using Altova XML Spy. http://quexml.sourceforge.net/sites/default/files/images/introducing_xml.pdf, last accessed 2015-05-07.

[167] Vladimir Gerasimov. to_ddi.xslt. https://github.com/p0rsche/equexmlyii/blob/master/quexml/to_ddi.xslt, last accessed 2015-05-10.

[168] Digital Curation Centre. Resources for digital curators. http://www.dcc.ac.uk/resources, last accessed 2015-02-21.

[169] Emma Tonkin. Persistent identifiers: considering the options. *Ariadne*, 56:8, 2008.

[170] Hans-Werner Hilse and Jochen Kothe. *Implementing persistent identifiers*. Consortium of European Reasearch Libraries, 2006.

[171] Roderic DM Page. Biodiversity informatics: the challenge of linking data and the role of shared identifiers. *Briefings in bioinformatics*, 9(5):345–354, 2008.

[172] Kevin Richards, Richard White, Nicola Nicolson, and Richard Pyle. A beginner's guide to persistent identifiers. Global Biodiversity Information Facility (GBIF), http://links.gbif.org/persistent_identifiers_guide_en_v1.pdf, last accessed 2015-02-02, 2011.

[173] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifier (URI): Generic Syntax. RFC 3986, January 2005. Updated by RFCs 6874, 7320.

[174] P. Leach, M. Mealling, and R. Salz. A Universally Unique IDentifier (UUID) URN Namespace. RFC 4122 (Proposed Standard), July 2005.

[175] Leo Sauermann, Richard Cyganiak, and Max Völkel. Cool URIs for the semantic web. Technical Memo TM-07-01, Deutsches Forschungszentrum für Künstliche Intelligenz, Kaiserslautern and Saarbrücken, Germany, 2007.

[176] Nikos Askitas, Janet Eisenhauer, Arofan Gregory, Rob Grim, Pascal Heus, Maarten Hoogerwerf, and Wolfgang Zenk-Möltgen. DDI 3.0 URNs and entity resolution. DDI working paper series: Best practice, Data Documentation Initiative, 2009.

[177] International DOI Foundation. Handbook: Registration agencies. `http://www.doi.org/doi_handbook/8_Registration_Agencies.html`, last accessed 2015-01-22, 2013.

[178] David Richter, Maria Metzing, Michael Weinhardt, and Jürgen Schupp. SOEP scales manual. SOEP Survey Papers 138, DIW-SOEP, Berlin, Germany, 2013.

[179] Boris Egloff, David Richter, and Stefan C. Schmukle. Need for conclusive evidence that positive and negative reciprocity are unrelated. *Proceedings of the National Academy of Sciences*, 110(9):E786, 2013.

[180] Martin Kroh. Short-documentation of the update of the SOEP-weights, 1984–2008. Technical report, DIW-SOEP, Berlin, Germany, 2009.

[181] Jan Goebel, C. Katharina Spieß, Nils RJ Witte, and Susanne Gerstenberg. Die Verknüpfung des SOEP mit MICROM-Indikatoren: Der MICROM-SOEP Datensatz. Technical report, DIW Berlin, German Institute for Economic Research, 2007.

[182] Survey of Health, Ageing and Retirement in Europe (SHARE). Methodological research. `http://www.share-project.org/methodological-research.html`, last accessed 2015-01-21.

[183] Herbert Brücker, Martin Kroh, Simone Bartsch, Jan Goebel, Simon Kühne, Elisabeth Liebau, Parvati Trübswetter, Ingrid Tucci, and Jürgen Schupp. The new IAB-SOEP migration sample: An introduction into the methodology and the contents. SOEP Survey Papers 216, DIW-SOEP, Berlin, Germany, 2014.

[184] Roger Koenker and Achim Zeileis. On reproducible economic research. *Journal of Applied Economics*, 24:833–847, 2009.

[185] Bruce D. McCullough. Got replicability? The *Journal of Money, Credit and Banking* archive. *Econ Journal Watch*, 4(3):326–337, September 2007.

[186] Christian Bizer. Expert report on linking data & publications, 2011.

[187] Adrian Jones. Breaking down the silos – data management across the enterprise. In *SAS Global Forum 2011: Data Integration*, volume 136, 2011.

[188] Matthias Schonlau, Martin Reuter, Jürgen Schupp, Christian Montag, Bernd Weber, Thomas Dohmen, Nico A. Siegel, Uwe Sunde, Gert G. Wagner, and Armin Falk. Collecting genetic samples in population wide (panel) surveys: feasibility, nonresponse and selectivity. *Survey Research Methods*, 4(2):121–126, 2010.

[189] Jürgen Schupp and Gert G. Wagner. Zum 'Warum' und 'Wie' der Erhebung von (genetischen) 'Biomarkern' in sozialwissenschaftlichen Surveys. SOEP-papers on Multidisciplinary Panel Data Research 260, DIW-SOEP, Berlin, Germany, 2010.

[190] German Internet Panel (GIP). Variable names – a guideline, 2014.

[191] Survey of Health, Ageing and Retirement in Europe (Share). Release guide 2.6.0 – waves 1 & 2. `http://www.share-project.org/fileadmin/pdf_documentation/SHARE_guide_release_2-6-0.pdf`, last accessed 2014-01-17, 2013.

[192] DDI Alliance. Representedvariable. DDI Field Level Documentation, `http://www.ddialliance.org/Specification/DDI-Lifecycle/3.2/XMLSchema/FieldLevelDocumentation/schemas/logicalproduct_xsd/elements/RepresentedVariable.html`, last accessed 2014-01-17, 2014.

[193] Uwe Jensen, Sanda Ionescu, Mari Kleemola, Agostina Martinez, Wendy Thomas, Mary Vardigan, and Wolfgang Zenk-Möltgen. Grouping of survey series using DDI 3. DDI Working Paper Series – Use Cases 6, Data Documentation Initiative, 2010.

[194] Thilini Ariyachandra and Hugh J. Watson. Which data warehouse architecture is most successful? *Business Intelligence Journal*, 11(1):4–6, 2006.

[195] Thomas Habing, Janet Eke, Matthew A. Cordial, William Ingram, and Robert Manaster. Developments in digital preservation at the university of illinois: The hub and spoke architecture for supporting repository interoperability and emerging preservation standards. *Library Trends*, 57(3):556–579, 2009.

[196] DDI Alliance. Concept. DDI Field Level Documentation, `http://www.ddialliance.org/Specification/DDI-Lifecycle/3.2/XMLSchema/FieldLevelDocumentation/schemas/conceptualcomponent_xsd/elements/Concept.html`, last accessed 2014-01-17, 2014.

[197] Arofan Gregory and Mary Vardigan. The web of linked data: Realizing the potential for the social sciences. NSF SBE 2020 paper 186, 2010.

[198] Tomi Kauppinen and Giovana Mira de Espindola. Linked open science-communicating, sharing and evaluating data, methods and results for executable papers. *Procedia Computer Science*, 4:726–731, 2011.

[199] Antoine Isaac and Ed Summers. SKOS simple knowledge organization system primer (W3C working group note 18 august 2009). http://www.w3.org/TR/skos-primer/, last accessed 2015-02-27, 2009.

[200] Marcus Eduardo Markiewicz and Carlos JP de Lucena. Object oriented framework development. *Crossroads*, 7(4):3–9, 2001.

[201] Martin Büchi and Wolfgang Weck. A plea for grey-box components. TUCS Technical Report 122, Turku Centre for Computer Science, 1997.

[202] German Socio-Economic Panel Study (SOEP). SOEP 2013 – documentation of person-related status and generated variables in PGEN for SOEP v30. SOEP Survey Papers 250, DIW-SOEP, Berlin, Germany, 2014.

[203] German Socio-Economic Panel Study (SOEP). SOEP 2013 – documentation of household-related status and generated variables in HGEN for SOEP v30. SOEP Survey Papers 252, DIW-SOEP, Berlin, Germany, 2014.

[204] Luigi Bellomarini, Nikolaos Zisimos, and Foteini Andrikopoulou. Validation & Transformation Language: VTL syntax. 2nd draft. Technical report, Banca d'Italia, 2014.

[205] David Martin, Mark Burstein, Jerry Hobbs, Ora Lassila, Drew McDermott, Sheila McIlraith, Srini Narayanan, Massimo Paolucci, Bijan Parsia, Terry Payne, Evren Sirin, Naveen Srinivasan, and Katia Sycara. OWL-S: Semantic markup for web services (W3C member submission 22 november 2004). http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/, last access 2015-02-27, 2004.

[206] John Gruber. Markdown. http://daringfireball.net/projects/markdown/, last accessed 2015-03-16.

[207] Tim T.Y. Lin. Scholarly markdown. http://scholarlymarkdown.com/, last accessed 2015-03-16.

[208] Github. Github Flavored Markdown (GFM). https://help.github.com/articles/github-flavored-markdown/, last accessed 2015-03-16.

[209] Rstudio. R markdown. http://rmarkdown.rstudio.com/, last accessed 2015-03-16.

[210] Google. R style guide. https://google-styleguide.googlecode.com/svn/trunk/Rguide.xml, last accessed 2015-03-16.

[211] Hadley Wickham. *Advanced R*. CRC Press, 2014.

[212] Yihui Xie. *Dynamic Documents with R and knitr*. CRC Press, 2013.

[213] Scott Chacon and Ben Straub. *Pro Git*. Apress, New York, USA, 2 edition, 2014.

[214] GitLab B.V. Gitlab Community Edition (CE). Software. https://about.gitlab.com/, last accessed 2015-03-19.

[215] Tom Preston-Werner, Chris Wanstrath, PJ Hyett, et al. Github. Software. https://github.com, last accessed 2015-03-19.

[216] Tom Preston-Werner, Rick Olson, et al. Gollum. Software. https://github.com/gollum/gollum, last accessed 2015-03-19.

[217] Tom Preston-Werner et al. Jekyll. Software. http://jekyllrb.com/, last accessed 2015-03-19.

[218] John MacFarlane et al. Pandoc. Software. http://johnmacfarlane.net/pandoc/, last accessed 2015-03-19.

[219] Robert Gentleman, Ross Ihaka, et al. R. Software. http://www.r-project.org/, last accessed 2015-03-19.

[220] CRAN. Contributed packages. http://cran.r-project.org/web/packages/index.html, last accessed 2015-03-19.

[221] Document Foundation. LibreOffice Calc. Software. https://www.libreoffice.org/discover/calc/, last accessed 2015-03-19.

[222] PostgreSQL. Software. http://www.postgresql.org/, last accessed 2015-03-23.

[223] Hadley Wickham. *R Packages*. O'Reilly, Sebastopol, USA, 2015.

[224] Apache Software Foundation. Apache maven project: Introduction to the standard directory layout. http://maven.apache.org/guides/introduction/introduction-to-the-standard-directory-layout.html, last accessed 2015-02-01.

[225] Rails Guides. Getting started with rails. http://guides.rubyonrails.org/getting_started.html, last accessed 2015-02-01.

[226] Rich FritzJohn. Designing projects. http://nicercode.github.io/blog/2013-04-05-projects/, last accessed 30 Jun 2014, 2013.

[227] Eric S. Raymond. *The art of Unix programming*. Addison-Wesley Professional, 2003.

[228] International Household Survey Network (IHSN). Microdata cataloging tool (nada). Software, http://www.ihsn.org/home/software/nada, last accessed 2015-03-27.

[229] Norwegian Social Science Data Services (NSD). Nesstar publisher. Software. http://www.nesstar.com/software/publisher.html, last accessed 2015-03-27.

[230] World Bank. Central microdata catalog. http://microdata.worldbank.org/catalog/, last accessed 2015-03-27.

[231] Metadata Technology North America. Openmetadata survey catalog. http://www.openmetadata.org, last accessed 2015-03-27.

[232] Jeremy Iverson and Dan Smith. Colectica. Software. http://www.colectica.com/, last accessed 2015-03-27.

[233] Algenta Technologies. Colectica user manual – for Colectica 4.1. http://cdn.colectica.com/Colectica%20User%20Manual.pdf, last accessed 2015-11-16, 2013.

[234] Alerk Amin, Michelle Edwards, Oliver Hopt, Jannik Jensen, Dan Kristiansen, Olof Olsson, and Joachim Wackerow. Questasy: Documenting and disseminating longitudinal data online using DDI 3. Ddi working paper series – use cases, no. 1, Data Documentation Initiative, 2009.

[235] Marika de Bruijne and Alerk Amin. Questasy: Online survey data dissemination using DDI 3. *IASSIST Quarterly*, 33(1):10–15, 2009.

[236] DDI Alliance. DDI Tools Catalogue. http://www.ddialliance.org/resources/tools, last accessed 2015-03-21.

[237] Mark Otto, Jacob Thornton, et al. Bootstrap. Software. http://getbootstrap.com/, last accessed 2015-03-23.

[238] jQuery Foundation. jQuery. Software. http://jquery.com/, last accessed 2015-03-21.

[239] Michael Bostock. Data Driven Documents (d3.js). Software. http://d3js.org/, last accessed 2015-03-21.

[240] SQLite. Software. https://sqlite.org/, last accessed 2015-03-21.

[241] Apache Software Foundation. Lucene. Software. https://lucene.apache.org/, last accessed 2015-03-21.

[242] Apache Software Foundation. Solr. Software. http://lucene.apache.org/solr/, last accessed 2015-11-10.

[243] DDI Alliance. DDI Lifecycle 3.2. Standard. http://www.ddialliance.org/Specification/DDI-Lifecycle/3.2/, last accessed 2015-04-13, 2014.

[244] Object Management Group. Unified modeling language (OMG UML), infrastructure. iso/iec 19505-1. Standard, Information technology - Object Management Group, 2012.

[245] Object Management Group. Unified modeling language (OMG UML), superstructure. iso/iec 19505-2. Standard, Information technology - Object Management Group, 2012.

[246] MySQL. Software. https://www.mysql.com/, last accessed 2015-05-13.

[247] Sam Goldstein et al. diffy. Software. https://github.com/samg/diffy, last accessed 2015-03-27.

[248] Lars Bertram, Anke Böckenhoff, Ilja Demuth, Sandra Düzel, Rahel Eckardt, Shu-Chen Li, Ulman Lindenberger, Graham Pawelec, Thomas Siedler, Gert G. Wagner, et al. Cohort profile: The Berlin Aging Study II (BASE-II). *International Journal of Epidemiology*, 2013.

Appendix

Appendix A

DDI on Rails – screen shots

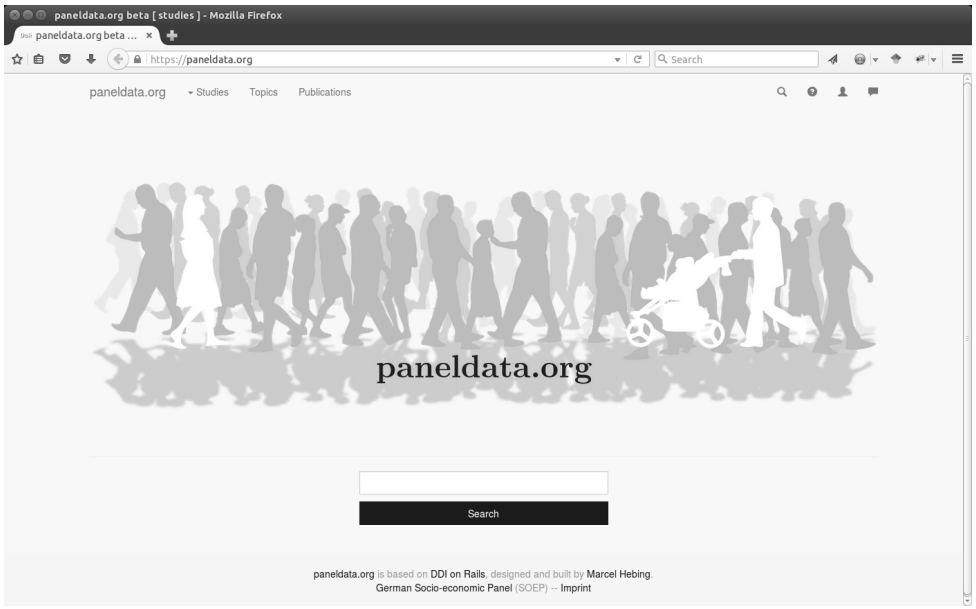

Figure A.1: The homepage of DDI on Rails provides direct access to the search interface, the study browser, the topics, and the publications. Further functionality becomes available after login. Screen shot of paneldata.org [41].

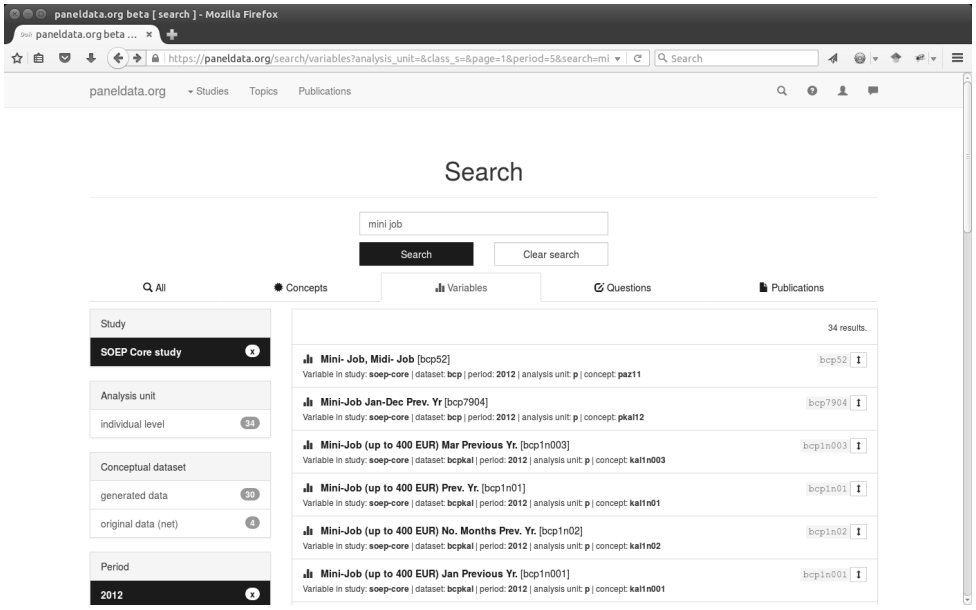

Figure A.2: The search interface allows to combine text search (text field on top) with facets (tabs below the text field and panels on the left) to specify search requests. Screen shot of paneldata.org [41].

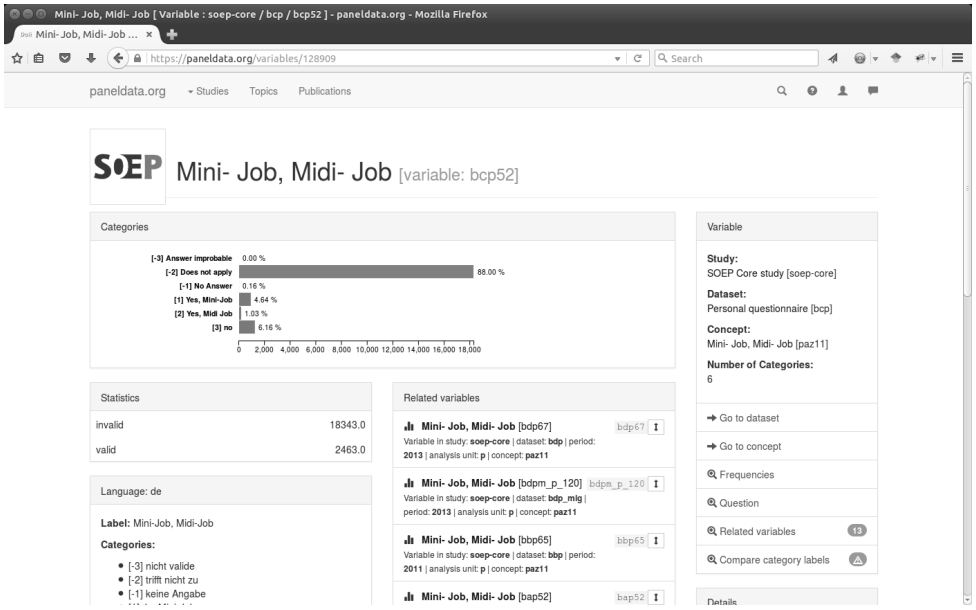

Figure A.3: The variable interface provides basic statistics (e.g., frequencies), details on the variable in the context of a panel study (e.g., links to related variables or comparison of categories over time as shown in picture A.4), and links to further material (e.g., concepts and questions). Screen shot of paneldata.org [41].

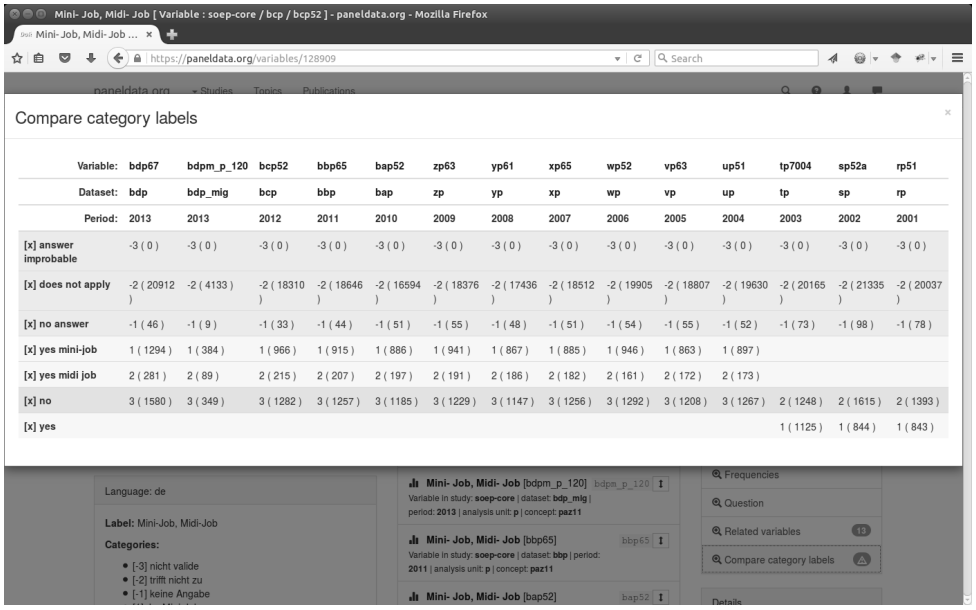

Figure A.4: The "label comparison" takes all variables from one study that are linked through a common concept. In the comparison table, the columns represent variables and the rows represent the variable categories. The individual cells first indicate whether a category was measured in a particular wave and, if so, how the category is coded in the data. Additionally, the number in brackets gives the corresponding frequency. This perspective supports panel researchers to identify inconsistencies over time and therefore potential problems in analysing a panel study. It also supports panel managers in ensuring the consistency of their data. In this screen shot, we can see, for example, that the "no" category is coded inconsistently over time—it is coded "2" for the years 2001–2003 and "3" for the years 2004–2013. Screen shot of paneldata.org [41].

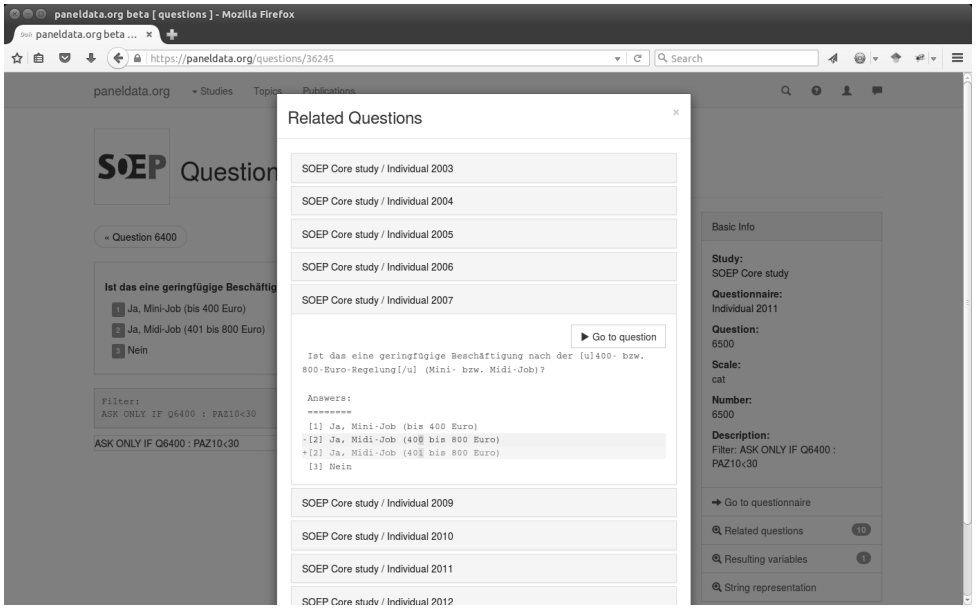

Figure A.5: Similar to the previous comparison of variable categories in figure A.4, concepts are also used to link and compare questions over time. After retrieving a set of related questions, a diff tool highlights changes in the questions. The example in this screen shot illustrates how even minor changes are identified and highlighted. In the user test, the use of a diff tool was considered to excel in illustrating the development of questions over time. More standardised approaches (e.g., classifying changes) failed because the relevancy of changes depends on the researcher's interests. Screen shot of paneldata.org [41].

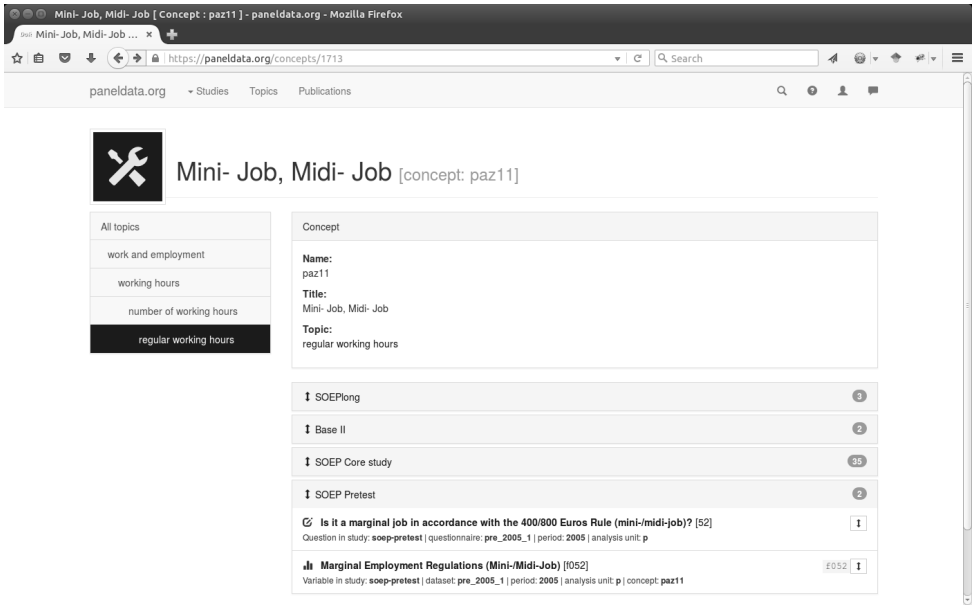

Figure A.6: While the previous examples (figure A.4 and A.5) illustrate how concepts are used to compare variables and questions for one study, the concept interface provides an overview of multiple studies that include measures for a particular concept. The two-sided arrows provide direct access to these elements (as shown for the SOEP Pretest study). This interface also includes the topics as an hierarchical structure on the left side. Screen shot of paneldata.org [41].

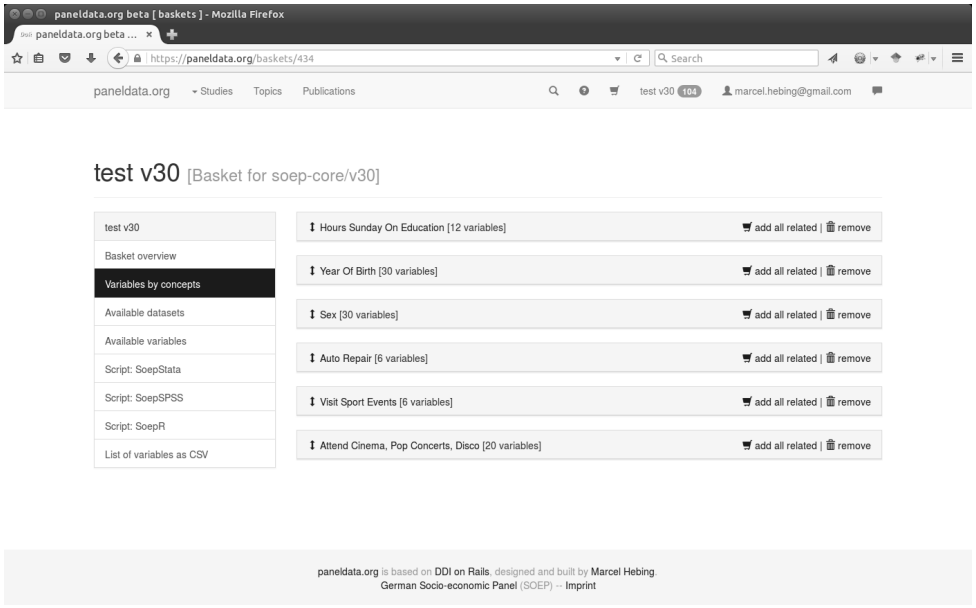

Figure A.7: Researchers can create an user account on paneldata.org and log into the system to create individualized baskets containing variables for one specific study release. Concepts are used to quickly add variables, which are related over time. Screen shot of paneldata.org [41].

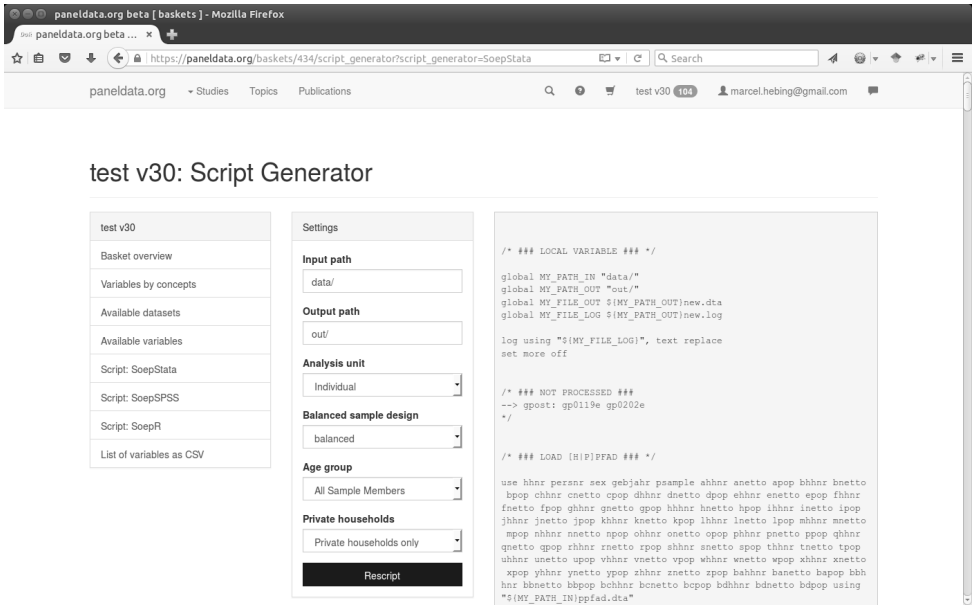

Figure A.8: The script generator enables researchers to export baskets to their preferred statistical packages. The script generator is of particular interest for researchers working with the cross-sectional version of SOEP Core. In this context, the generated code automatically selects related variables over time from more than 200 datasets and combines the data into a single dataset (wide format). Screen shot of paneldata.org [41].