A Generic Quality-Focused Data Management Process that distinguishes between semi-structured Research Data, Data Models and Data Transformations

Process Definition

Developed as part of the BMBF-funded project KONDA

KONDA - Continuous quality management of dynamic research data on objects of material culture using the LIDO standard

Authors	Markus Matoni, Arno Kesper, Viola Wenz, Gabriele Taentzer
Date	03.07.2023
Version	0.1
Status	not published
Release	04.07.2023

Bundesministerium für Bildung und Forschung
Philipps Universität Marburg
GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN
NIEDERSÄCHSISCHE STAATS- UND UNIVERSITÄTSBIBLIOTHEK GÖTTINGEN | SUB

Abstract

Research data, associated data models and data transformations are often subject to continuous change and high-quality expectations. Hence, ensuring quality is a particular challenge in research and quality assurance is key. Data management processes are known and increasingly used in practice, but, like the ISO Standard 9001[1] and the GFBio Life Cycle [RFII 2019], they handle Quality Assessment as independent tasks and activities. As a result, data quality is not continuously ensured in data management processes since data quality is not the focus. Furthermore, data quality is closely related to data model quality and data transformation quality, and data quality management processes implicitly address data models and transformations at best. This can indirectly reduce data quality, as poor quality of data transformations and models directly impacts data quality. Therefore, this document defines a quality-focused data management process (QDMP) that distinguishes between semi-structured research data, data models, and data transformations. A meta model fully structures and complements the process with activities, tasks, techniques, artifacts, capabilities, and roles. These key concepts indicate the interrelationships and dependencies of activities and, thus, between data, data models, and data transformations. The process results from a three-year research called KONDA[2], focusing on the quality of research data.

[1] https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/publication/10/03/PUB100373.html
[2] https://zenodo.org/communities/konda-project

Table of Contents

Introduction

The digital transformation of our society is a constant challenge, and almost every action in the digital world generates data. To use the available data effectively, it must be of high quality. Today, data is often subject to constant change, especially research data, since uncertain and vague knowledge can be confirmed over time or because there is a high demand in research for greater data integration. Ensuring data quality is a particular challenge in research and key for high-quality scientific work. To enable high-quality research data, our goal is to assure data quality continuously through the whole data life cycle by applying quality-focused data management processes. In this article, we focus on semi-structured, linked data, which are often used in research areas such as cultural heritage, literature, and biodiversity. Data quality problems often originate from quality problems in corresponding data models and data transformations. Thus, to enable high-quality research data, a key challenge is to apply quality-focused processes not only for data but also for data models and data transformations. This document considers the systematic quality assurance of data, associated data models, and data transformations.

Data models and data transformations play an essential role in the data life cycle. Data models provide structure and rules and thus directly influence the quality of data, e.g., through controlled values or mandatory elements. Many public data are created by data transformations. Poor quality of data models and data transformations can lead to information loss and incorrect information. To assure the quality of data, it is not sufficient to consider only the data itself since the data models' and data transformations' quality significantly impact data quality. The literature lacks quality management processes considering the dependencies between data, data models, and data transformations and their qualities. One of the reasons why research data may have varying quality is that there are a variety of data models and data transformations. Thus, many data models and data transformations remain local in research organizations and are not subject to quality assurance. This makes it all the more essential to define quality-oriented data management processes not only for data but also for data models and data transformations.

However, the processes defined and applied by organizations either consider data, data models, and data transformations separately or data models and data transformations are only implicitly mentioned in processes for data. For example, data models are only mentioned in data management processes when choosing the structure and data model for data. This corresponds to data-focused approaches like the GFBio Data Life Cycle [RFII 2019]. Data transformations are not mentioned in data management processes at all. Thus, if applicable, only their requirements and problems are implicitly mentioned in the area of data acquisition. However, specific activities of data transformations are not mentioned, nor is it specified how these problems came about through data transformations. Fundamentally, data, data models, and data transformations are to be described in coequal processes since both data model quality and data transformation quality have a direct impact on data quality. In turn, data transformation quality also depends on source data and data models. It is essential that these and other types of dependencies are considered in combination, that these three scopes are defined at the same level, and that the intended sequence of activities of these three scopes is precise.

Data management processes (DMP) and quality assurance techniques are known in many research organizations. Nevertheless, there are several DMPs, such as the ISO Standard 9001, that do not cover quality assurance tasks within the defined data management activities. Thus, quality assurance tasks are considered as independent and detached tasks, as in the GFBio Life Cycle [RFII 2019]. In fact, quality assurance should be part of the different phases of a DMP and thus be continuous and linked to the respective tasks and techniques. Approaches, practices, or even standards that assure quality for some tasks already exist and can be assigned to particular tasks. For example, the ISO/IEEE 830-1998 standard[3] exists for specifying requirements for software and is essential for defining requirements for data transformations.

Like checking for plausibility in dates when creating data - However, these methods have yet to be assigned to particular activities of a data management process (DMP) and applied jointly.

Research organizations and individual researchers operate in different domains and have various research methods to address specific research questions. This results in different demands and requirements for data quality. Therefore, a process is essential that is both generic to be adaptable to different use cases and technologies as well as continuous to comprehensively assure the quality of data, data models, and data transformations in organizations' life cycles. Thus, this document defines a quality-focused data management process that addresses data quality as well as the quality of data models and data transformations. The goal is to make the dependencies and interrelationships explicit. Thus, this process identifies quality-related activities, tasks, techniques, artifacts, and roles to support research organizations and individual researchers in ensuring that data are of high quality.

This work provides the following contributions:
(1) We define a *quality-focused data management process that explicitly distinguishes between data, data models, and data transformations*. The key deliverable is the definition of the process activities that target the quality assurance of data, data models, and data transformations. All these activities are interrelated, and the dependencies and interrelationships are made explicit by defining these activities related artifacts within and across these scopes.
(2) The process presented is *quality-focused* in that it defines data management activities to ensure the quality of data, data models, and data transformations. These activities focus on tasks and techniques necessary to reach high quality of the activities' output artifacts.
(3) The process approach enables organizations to plan their activities to ensure that these activities are adequately resourced and that opportunities for quality assurance are implemented. This QDMP is designed to be *generic* in that it is not intended to be tailored to a particular research domain but to be instantiable and adaptable for different use cases. These use cases should specify specific tasks and techniques based on the generic process to ensure quality.
(4) The process is not only domain-independent and technology-independent, but also continuous, as it is intended to accompany the entire life cycle of research data, data models, and transformations.

[3] https://standards.ieee.org/ieee/830/1222/

This document is structured as follows: Section "*Scope"* defines which type of data the process addresses. The section "*Terms and Definitions*" describes the basic notions since many approaches to DMP and QMP do not define the contextually relevant terms. However, these are essential for defining a quality-focused data management process. The section "*Methodical Approach"* explains how the requirements for this process were gathered and how the process was developed. The section "*Related Work"* describes related work and shows where there are overlaps and differences with this process. The section "*Process Definition"* presents the actual process. It contains a subsection that describes how the chapter is structured and understandable to read and a subsection that defines the meta model of the process. Furthermore, the process itself is defined using all the components of the corresponding meta model: activities, tasks, techniques, skills, roles and artifacts.

The Generic Quality-Focused Data Management Process that distinguishes between semi-structured Research Data, Data Models, and Data Transformations will be abbreviated as QDMP in the following.

Scope

The process of this document is designed generic and intended to apply to any size or type of organization that implements and maintains semi-structured research data. Since research data include diverse and varying forms of data that lead to different levels of data quality, ensuring the quality of these data is correspondingly challenging and essential. Organizations that manage research data often face the requirement to produce high-quality data despite the heterogeneity, dynamism, and uncertainty associated with research data. Depending on the form of the data, the languages, techniques, and tools for processing the data may differ. We will focus on semi-structured data in this document, as this is the predominant form of data used in many research domains, like the cultural heritage domain. In doing so, we have based our work on experience, problems and from these domains. However, the process is designed to be generic. Further evaluation will show to what extent the process can be instantiated to other domains and applied to other types of data.

To assure the quality of data, it is not sufficient to consider only the data itself. Data models and data transformations play an essential role in the data. Therefore, we define a quality-focused data management process that distinguishes data, data models, and data transformations. The focus lies on inherent data, data models, and data transformations. This excludes for instance data processing systems such as database systems, information systems, or search engines. By presenting this QDMP a form of data management is assumed. But, unlike traditional DMPs, this process focuses on activities to assure the quality of data, data models, and data transformations. These activities focus on tasks and techniques necessary to reach high quality of the activities' output artifacts.

Methodical Approach

Given a large variety of quality problems that may have various concrete forms, a process-oriented approach is promising for permanently structuring activities and workflows

independent of concrete technologies and formats. For this approach, we have chosen a requirements-driven method to define the process by identifying quality problems of semi-structured research data, data models, and data transformations as well als requirements for this QDMP.

First, we empirically and analytically captured quality problems structurally, classified them and grouped them by the categories. This resulted in a catalog [Kesper et al. 2023] as a comprehensive specification of 49 data quality problems, 16 data model quality problems, and three data transformation problems, through structured capturing of various aspects per problem, such as its impact on data quality and possible causes. These quality problems serve as preliminary work for this process, and we consider them as quality assurance requirements for this document. The catalog approach is based on an empirical survey and an explorative analysis of data, data models, and data transformations. The empirical survey is based on a community workshop with 19 experts and six expert interviews. We specifically selected experts from acquisition, modeling, management, and the use of research data of, in particular, cultural heritage objects.

- According to Fouché et al. [Fouché et al. 2010], we designed the community workshop exploratively as a world café from qualitative social research with moderated meetings on six different topics, open questions, discussions, and evaluations based on the anonymous protocols and the flip charts of the sessions.
- According to Bogner et al. [Bogner et al. 2014], the expert interviews were conducted from qualitative social research with individually customized guides as questionnaires, digital pseudonymized recording, transcription, and evaluation.
- The explorative data analysis is based on data from the Centre for Collection Development of Göttingen University[4] and the Deutsches Dokumentationszentrum für Kunstgeschichte - Bildarchiv Foto Marburg[5]. These semi-structured research data sets consist of nearly 30000 data records from the Centre for Collection and over 800000 data records from Foto Marburg, with additional 360000 records containing supplementary information. The underlying data models of these data are LIDO[6] and MIDAS[7]. Both models are object-based XML schemas representing data models from semi-structured research data. We examined these two data models for quality problems and compared the findings with the quality problems of related data models for semi-structured data. We also considered data transformations to LIDO in this context since LIDO is a standard exchange format in cultural humanities. Thus, data transformation quality problems are representative of semi-structured data.

Based on this empirical and analytic evaluation of quality problems, we analyzed existing quality problems in processes, identified preventively and retrospectively improvement

[4] Centre for Collection Development of Göttingen University. https://www.uni-goettingen.de/en/440706.html (Accessed 2020-08-05)
[5] Germany's documentation center for art history, Deutsches Dokumentationszentrum für Kunstgeschichte – Bildarchiv Foto Marburg. https://www.uni-marburg.de/de/fotomarburg (Accessed 2020-08-04)
[6] Erin Coburn, Richard Light, Gordon McKenna, Regine Stein, Axel Vitzthum. „LIDO (Lightweight Information Describing Objects)". http://network.icom.museum/cidoc/working-groups/lido/. (Accessed 2020-08-04)
[7] Laupichler, Fritz, MIDAS, HIDA, DISKUS – was ist das?, 10.11588/heidok.00006198, (Accessed 2020-08-04)

methods, and conceived needs for a QDMP. Further, we organized a second workshop to consolidate the quality problems of actual processes and concretize requirements for target processes. This workshop involved experts in data modeling, data collection, data analysis and data transformations. To achieve a high level of interaction and productive discussions, even virtually, we designed four sessions: a) Classification of data quality improvement steps, b) Transferability of procedures for the analysis and improvement of software (model) quality to data models, c) Improvement of data model quality by explicit modeling of uncertainties, d) Improvement of the process quality of data transformations.
Each session began with a keynote presentation by a selected external expert. This was followed by a virtual session based on the World Café concept with open questions. The participants were asked to identify problems and develop solutions collaboratively. The online whiteboard tool was used for this purpose so that users could exchange ideas and work interactively. Questions that were particularly relevant for this document were: How can processes improve data, data models, and data transformation quality? What problems exist with actual processes? How can we optimize these processes? As a result of this workshop, we identified process steps necessary for data, data model, and data transformation quality improvements. We further identified roles, tools, and techniques and assigned them to these steps. These steps provided a basis for defining the activities in the QDMP.

Based on this preliminary work, we analyzed the identified quality problems, improvement methods, and requirements for a quality-focused data management process. More precisely, this involved determining which steps, tasks, and techniques are essential to ensure data, data models, and data transformation quality independently of each other. As a result and one of the key findings of this work, we have identified three data management processes consisting of activities. These processes each represent the target process from a quality perspective for the scopes data, data models, and data transformations. Thereby, the activities are linked scope-internally to define dependencies within the scope and cover the entire life cycle. In other words, the interrelationships of the data quality activities have been made explicit by one or more links.

Through the empirical and analytical findings of the preliminary work, the causes of multiple quality problems show that data, data model, and data transformation quality problems may affect each other. Therefore, it was essential to investigate all activities' dependencies, define a unified process representing all quality-relevant activities, and make the interrelationships explicit. Further, we defined the activities since they contain essential process information and assigned quality-related tools, techniques, and artifacts. To obtain such a quality management process definition, its implementation should be defined systematically and methodically by a **meta model**. Therefore, we have defined a meta model to denote the set of elements necessary to describe a quality-focused data management process for this work. This meta model includes not only the activities it contains, but also the tasks to be performed, the artifacts to be produced, the techniques used, the roles involved in the organizations, and the relationships between these concepts.

Related Work

The approach of this process and its meta model takes up several related works. Since our considerations of related work cannot be complete, we focussed on the well-accepted ISO

9001 standard for quality management and the common research approach of the RFII. In defining the metamodel, we focused on work from the field of method and software engineering, in particular, the work of Engels and Sauer as well as of Brinkkemper.

The **ISO Standard 9001**[8] promotes creating and adopting a process approach when "developing, implementing and improving" a quality management system. The standard, therefore, presents strategic recommendations to consistently provide products and services that meet the requirements of customers and regulators. The approach is designed on an even more general and organizational level to document principles for quality management measures. These principles are intended to facilitate mutual understanding on a stakeholder level. However, the object of this process is not information and data but products and services. Part of this process is the PDCA-cycle (Plan, Do, Check, Act). Plan, Do, Check, and Act can be seen as activities. The Plan-Do-Check-Act cycle is a model for implementing change. It is a key enabler for continuous process improvement and essential to this QM process. PDCA is a simple four-step method that organizations can use to avoid recurring errors and improve processes. This idea is one of the foundations of our process.

GFBio [RFII 2019] presents a data management process for research, designated as a data life cycle. This life cycle consists of steps that can be compared to activities. It supports every data producer and re-user to manage data, partially ensuring data quality. The specified life cycle guides the research organizations and is part of the project service. However, the process is not based on an explicit meta model and lacks defined techniques and dependencies between activities. In particular, activities of data models and data transformations are not explicitly considered.

Brinkkemper [Brinkkemper 1996] early mentioned that software engineering methods should be developed systematically and introduced the term method engineering. Besides defining the key terms related to that topic, he presents a concept for situational methods. Situational methods face the project life cycle by defining the principle of how methods and techniques interact in a project environment adaptly. Basically, they define a method that describes this principle. However, this method is very generic and can only be used for this QDMP to a limited extent. Since we need a concrete meta model for this process, we base our approach on the work of Engels and Sauer, who, in turn, rely on Brinkkemper's method.

Engels and Sauer [Engels and Sauer 2010] present *MetaME,* a meta method for modeling and tailoring software engineering methods. We adapt parts of this approach to define a data engineering method, namely the quality-focused data management process, which explicitly uses data models and data transformations. *MetaME* is described as a class diagram and defines the relationships between the core elements such as activity, task, role, artifact, technique, etc. The relationships are briefly described, but they miss to define the elements themselves, so it remains vague how the elements like task and work are defined. The entire meta model is also pervasive, containing project-related elements, such as milestones, project phases, domains, and disciplines. Because of the vague definition, we could not adapt these elements to our process. While several elements, such as process, activity, and task, correspond to elements in our meta model, we only adapt the element technique, along with the subdivision into tool, guidance, and utility. Due to the lack of definitions for these

[8] https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/publication/10/03/PUB100373.html

elements, the delineation between these elements remained unclear. Thus, we have redefined these elements for this QDMP meta model.

In our analysis, we have observed why the mentioned processes are not applicable for quality management for data, data models, and data transformations of semi-structured research data. The most obvious point is the **lack of meta models** for these processes that specify how to develop a quality management process based on the meta modeling. In other words, how an organization should instantiate the process. Secondly, although the approaches offer some aspects of data models and data transformations, at least on a high level of abstractions, they **lack the definition of specific data model and data transformation activities** to ensure high quality of those. As the analysis of data quality problems shows (see the catalog of quality problems), many causes of data quality problems are in the data models and data transformations problems. In other words, data model problems and data transformation problems directly affect the quality of data. It is, therefore, necessary to consider activities of data, data models, and data transformations together in a quality-focused data management process.

Terms and Definitions

In order to manifest a common understanding of the fundamental concepts and terms that are used in this document, the following terms and definitions apply.

Process: is a "set of interrelated or interacting activities that transforms inputs into outputs". (ISO 9000:2005)

Activity: is a “set of cohesive tasks of a process”. (ISO/IEC/IEEE 15288:2015)

Data: is a multifaceted term with widely separated ideas and concepts behind it. The ISO ISO25012 standard defines *data* as a “reinterpretable representation of information in a formalized manner suitable for communication, interpretation, or processing."

Data Model: A data model is a “graphical and/or lexical representation” model “specifying the properties, structure, and inter-relationships” (ISO/IEC 11179) of data of a domain of interest. In the context of relational data or XML data, a data model is also referred to as a schema.

Data Transformation: A “process which creates new data from an original source”. (ISO 5127:2017)

Data Quality: is a multi-dimensional construct and is still commonly conceived with a simple definition “fit for use”. Without contradicting this simple definition, we consider data quality as the “degree to which the characteristics of data satisfy stated and implied needs when used under specified conditions.” (ISO 25012)

Data Model Quality: There is no common standard definition for data model quality. We consider data models as a special kind of data. Accordingly, we also consider data model quality to be a special kind of data quality.

Data Transformation Quality: There is no common standard definition for data transformation quality. We consider data models as a special kind of data. Accordingly, we also consider data model quality to be a special kind of data quality.

Data Quality Management: “coordinated activities to direct and control an organization with regard to data quality” (ISO 8000-1:2020)

Research Data: “is data collected, observed, or created, for purposes of data analysis to produce original research information and results”. (ISO 5127:2017) It is based on information as a result of scientific work and generated in the course of scientific projects, e.g. through digitization, source research, experiments, measurements, surveys, or interviews. Characteristics of research data are the variety, dynamics, uncertainty, and the effort for preservation which leads to another level of data quality.” [Ray 2014, RFII 2019]
Research data include, but are not limited to, measurement data, laboratory values, audiovisual information, texts, survey data, objects from collections, or samples that are created, developed, or analyzed in the course of scientific work. [Deutsche Forschungsgemeinschaft 2015]

Semi-structured data: Abiteboul et al. [Abiteboul 1996 state that “the data resides in different forms, ranging from unstructured data in file systems to highly structured in relational database systems.” Semi-structured data “is (from a particular viewpoint) neither raw data nor strictly typed, i.e., not table-oriented as in a relational model or sorted-graph as in object databases.” Structured data is always accompanied by a data model explicitly defining its structure. In the relational model, a data structure typically consists of interrelated data entities with attributes. Each attribute is named and has a data value. In contrast, unstructured data, such as text, images, and video, exist without their structure being explicitly represented, either internally or externally. Semi-structured data is represented using a data description language such as XML that pre-structures the data. Therefore, Ambika defines semi-structured data as “a combination of structured and unstructured data and shares characteristics of both” [Ambika 2020]

Skill: An “ability to apply knowledge and experience to complete tasks and solve problems”. (ISO 14731:2019)

Role: is a function or position that specifies responsibilities (ISO 81001-1:2021), (ISO 18435-1:2009) and (ISO/IEC 11179-6:2015). In this document, a role is assigned to one or more skills.

conNextActivity (connection to next activity): A connection to one or more activities that should follow in order. Hence, the following activities should be performed next to ensure the data, data model, and data transformation quality.

concurrentActivity: A concurrent activity is an instance of an abstract activity that can occur concurrently with other activities. A concurrentActivity has one or more activities that are concurrent. These are listed in the QDMP under *hasConcurrentActivities.*

Task: An atomic work unit to achieve a goal as part of an activity. In this process, tasks relate to quality assurance, not data management. For this, a task is distinguished into constructive and analytic QA tasks.

Constructive QA Task: ensures that “mistakes are minimized during the creation of a work product.” (here: output of activity). “That is, they prevent issues from being introduced.” [Aurum and Wohlin 2005]

Analytic QA Task: is “performed on” a “completed artifact” (here: output of activity) “with the aim to detect issues.” [Denger and Olsson 2005]

Technique: is a “way that a method is realized, or implemented.” (ISO/IEC 16500-8:1999) In this process, techniques relate to quality assurance, not data management. For this, a technique can be composed of tools, utilities, and guidance:

- Tool: A software that is used to assist in and automate parts of the process of management of software assets (ISO/IEC/IEEE 24765:2017). Here we define tools as software that is used interactively by the user.
- Utility: A script “that can be called by name [...] to perform a specific task or related set of tasks” (ISO/IEC/IEEE 9945:2009). Included are those that query parameters or require configuration without influencing the content of the final result.
- Guidance: A dialogue element or documentation that aids “the users in achieving their intended results.” Here, the intended result is creating the artifact by performing the task. “Guidance can aid users in discovering the capabilities of a system, enable the users to generate a plan for accomplishing their goals, assist the users in accomplishing a goal, or help the users to manage error situations.” (ISO 9241-13:1998)

Artifact: is a “work product” that is “produced and used during a project to capture and convey information”. (ISO 19014-4:2020) For this document it is either necessary for a particular activity as an input or output that is produced in another activity.

Use Case: A “sequence of actions that an actor (usually a person, but perhaps an external entity, such as another system) performs within a system to achieve a particular goal” (ISO 17185-1:2014). Here, the actions are grouped into activities and the actors are entities grouped into roles.

Process Definition

This chapter defines the *generic quality-focused data management process* that distinguishes between data, data models, and data transformations. The management of data, data models, and data transformations is defined and visualized in separate processes. Data, data models, and data transformations are defined and visualized as individual processes and together in an integrated process to make dependencies explicit. In contrast to data management processes, we focus on *quality assurance* (QA) tasks assigned to this process's activities. Since pure data management processes and quality management processes have different purposes, the focus here is on linking data management tasks with specific quality goals leading (*ConstructiveQATasks*) or (*AnalyticQATasks*) that check the

quality of the resulting artifacts. In principle, the process remains generically defined. Thus, it is domain and technology independent but instantiable and adaptable to different use cases.

How To Read

This process definition is structured as follows:

- **Meta Model:** the meta model of this process including the terms and their relation that are needed to define the process.
- **Visualization:** The visual overview of the Quality-Focused Data Management Process that distinguished between Data, Data Models and Data Transformations
- **Activities**: described, grouped by data, data models, and data transformations, and sorted by their order in the process, starting with requirements analysis
- **Tasks** (QA tasks that are used in activities): described, sorted alphabetically, and linked via the activities
- **Techniques** (Techniques that are used in QA tasks): described, sorted alphabetically, and linked via the tasks
- **Skills** (that are used in tasks): described, sorted alphabetically, and linked via the techniques
- **Roles**: described, sorted alphabetically, and linked via the skills
- **Artifacts**: described, sorted alphabetically, and linked via the input and output of the activities

Meta Model

In this work, we use the term *meta model* to denote the full set of terms needed to describe a quality-focused data management process. This does not only cover its contained activities but the tasks that need to be performed, the artifacts that are to be produced, the techniques that are applied, the roles that participate in the organizations, and relationships between these concepts.

To establish processes, their development should follow a structured and methodical approach. This is the objective of method engineering [Brinkkemper 1996]. Method engineering is a discipline that focuses on “the design, construction, and evaluation of methods, techniques and support tools for information systems development" [Brinkkemper 1996]. Originating in the field of *information systems* during the 1990s, *software engineering* later embraced *method engineering* [Engels and Sauer 2010]. Numerous software engineering methods and processes have been proposed and utilized for various purposes, such as the Rational Unified Process (RUP) [Shuja and Krebs 2008], agile methods like SCRUM [Schwaber and Beedle 2002], and many others. However, these standardized approaches often prove too generic to be directly applicable and must be adapted to specific domains and contexts before they can be effectively implemented [Engels and Sauer 2010].

The meta model of this work is designed to support the definition of a quality-focused data management process for semi-structured research data, data models, and data transformation, as well as tailoring this process for particular research domains and technologies. *Figure 1* presents this meta model specifying the elements and how they

interrelate. It builds in certain parts on the meta model presented by Gutzwiller in 1994 [Gutzwiller et al. 1994]. Gutzwiller defines five key elements that are part of this meta model: *activity*, *role*, *work product* (here artifact), and *technique*. The element *technique*, along with the subdivision into *tool*, *guidance*, and *utility*, is based on Engels and Sauer [Engels and Sauer 2010] and has been adopted for this meta model. Due to the lack of definitions for these elements, the delineation between these elements remained unclear. Thus, we have redefined these elements (see *section* Terms and Definitions) for this meta model.

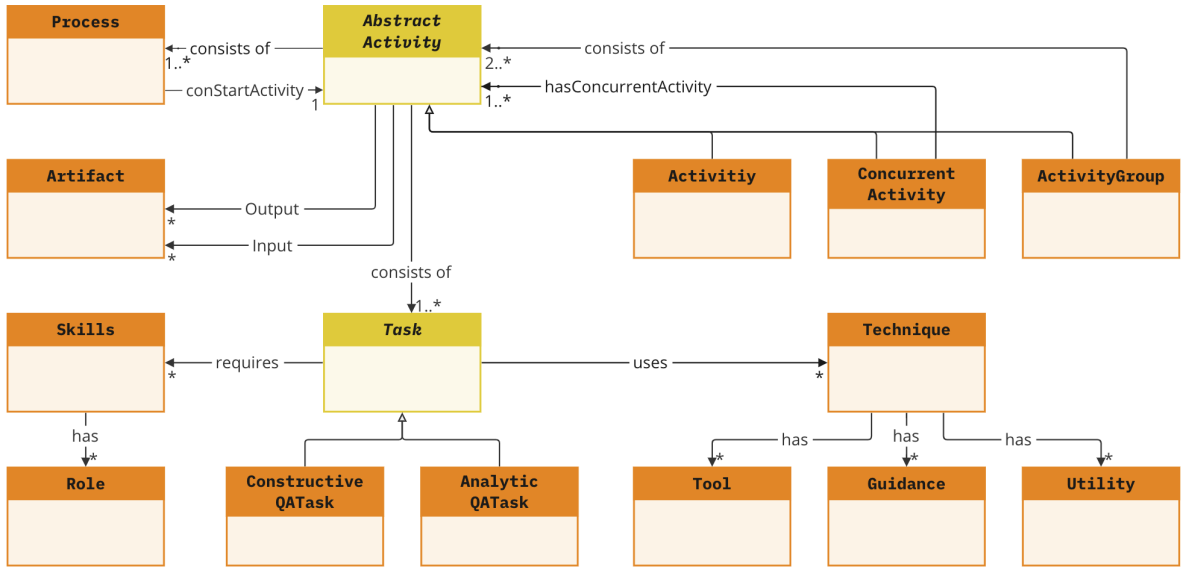

Figure 1: UML class diagram of Meta Model for Quality-Focused Data Management Processes for Data, Data Models, and Data Transformations. Orange colored classes are UML Classes. Yellow colored classes are *abstract classes*.

The upper half represents the core concept of a *process*. A *process* consists of *activities*. Activities can be defined in three different ways: As an ordinary activity, as an *activity* that is incidental to other *activities*, and as a *group* that includes multiple *activities*. All of these *activities* can have inputs and outputs in the form of *artifacts,* and they can contain quality assurance *tasks*. Here, tasks are either *constructive* or *analytical QA tasks*. The bottom left corner shows that tasks require *skills*, which in turn are assigned to *roles*. At the bottom right corner, techniques are needed to perform the *task*. These *techniques* distinguish between *Tool*, *Guidance,* and *Utility*. These include *guidance* on how to perform a *task*, *tools* that support users while performing a *task*, and *utilities* that only need to be executed by the user to perform the *task* automatically.

Process and Visualization

This section presents the integrated Quality-Focused Data Management Process that distinguishes Data, Data Models, and Data Transformations. This visualization shows the relations and dependencies between the activities. It is the most important point of contact from the perspective of entry into the process, as it provides an overview of all existing activities and their interrelationships.

The defined process is formally described as a graph where the nodes represent activities and the edges represent the connections between those activities. This process is generic since it is applicable to any context of data. Hence, each use case is part of the process as it represents a subgraph of the process. Each use case defines a subset of activities with a fixed sequence and, consequently, a start and end activity. Therefore, the generic process does not have a fixed start or end activity in practice. However, projects that start without any data, data models, or data transformations, need a formal starting activity. Therefore, there is a formal starting activity for this process.

Although the process is visualized as a graph to simplify the representation, it contains all use cases that address data, data models and data transformations of semi-structured research data. Use cases can run in parallel and at different times, but they can also influence each other. To explain this, we would like to illustrate a simple example. Say, the activity (Data) Design is part of a use case U1 that contains the choice of a data model. At the time of this choice, the data model has already been created. Creating a data model is a separate use case, U2, whereby the activities of this use case have already been run through. Accordingly, use case U1 and use case U2 occurred at entirely different times.

The defined process is formally described by formulating the respective components in textual terms. To visualize the textual description, we distinguish between a temporal and a causal representation. To effectively handle data, data models, and data transformations, it is crucial to follow a specific chronological sequence. Hence, it is important to indicate the activities within these three processes that should occur concurrently or sequentially. The artifacts generated by each activity serve as inputs for other activities. To visualize the dependencies between activities, a causal representation is essential.

Temporal Representation

The temporal visualization is an activity diagram according to the UML 2.2 specification[9], where the activity nodes represent activities and the activity edges represent the connections between those activities. Multiple connections between two activity nodes are modeled as decision and merge nodes as well as fork and join nodes.

To identify the relationships and temporal dependencies between activities of data, data models, and data transformations, it is useful to model the quality-focused data management process that distinguishes between data, data models, and data transformations separately. *Figure 2* therefore gives an overview of the activities in a quality-focused data management process.

[9] OMG Unified Modeling LanguageTM (OMG UML), Superstructure. https://committee.iso.org/files/live/users/fh/aj/aj/tc211contributor%40iso.org/files/Presentations/2010-12%20Canberra/09-02-02_Superstructure.pdf

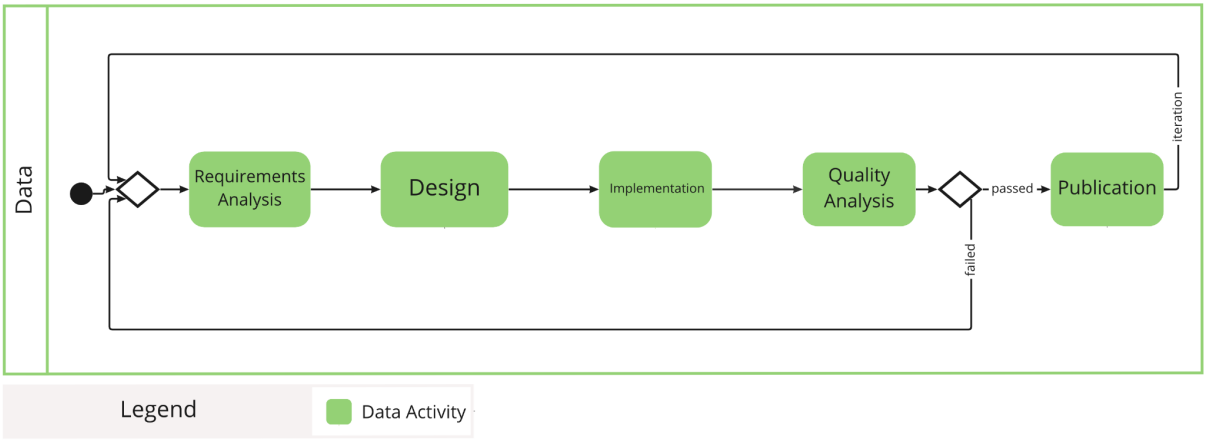

Figure 2: Generic Quality-Focused Data Management Process for Data: Visualization of the activities and their temporal dependencies in the generic Quality-Focused Data Management Process for Data. UML Activity Diagram, supplemented with green activities for mapping to data.

The green color highlights these data activities. Connections between activities are either should-connections or or-connections when there are decision nodes. There is one decision node. To be precise, the activity "quality analysis" is followed by a decision. Depending on the result of the quality analysis, the requirements analysis or/and the publication should follow. The other activities are connected by means of basic should-connections and should be completed one after the other. The *section Activities* defines which tasks, inputs, and outputs the activities should include. At this point, only the activities themselves and their temporal relationships are presented.

Analogous to the activities of data, the activities of data models are also subject to their own quality-focused data management process. It is therefore useful to initially consider their activities as a whole and define connections between them. *Figure 3* presents these data model activities in orange and the connections between them. It is evident that the number and the name of activities are identical to those of data.

The activities of data transformations are equivalent to the ones of data models. *Figure 4* presents the activities of data transformations in blue, which are also subject to their own quality-focused data management process.

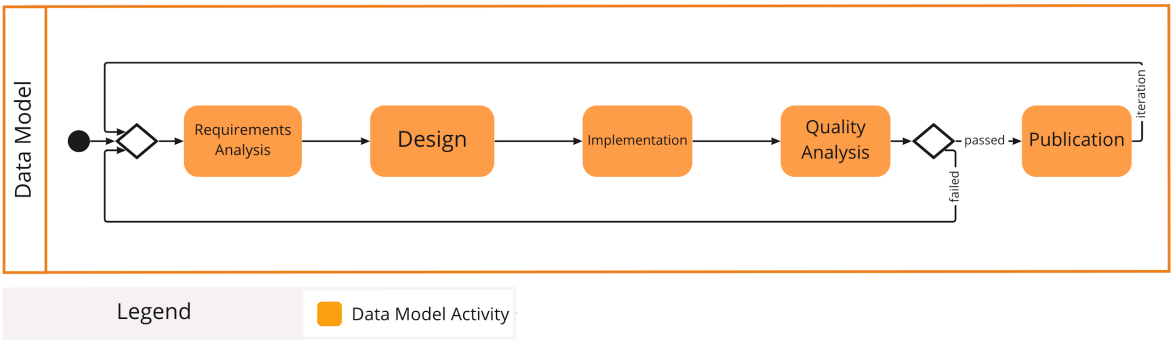

Figure 3: Generic Quality-Focused Data Management Process for Data Model: Visualization of the activities and their temporal dependencies in the generic Quality-Focused Data Management Process for Data Models. UML Activity Diagram, supplemented with orange activities for mapping to data models.

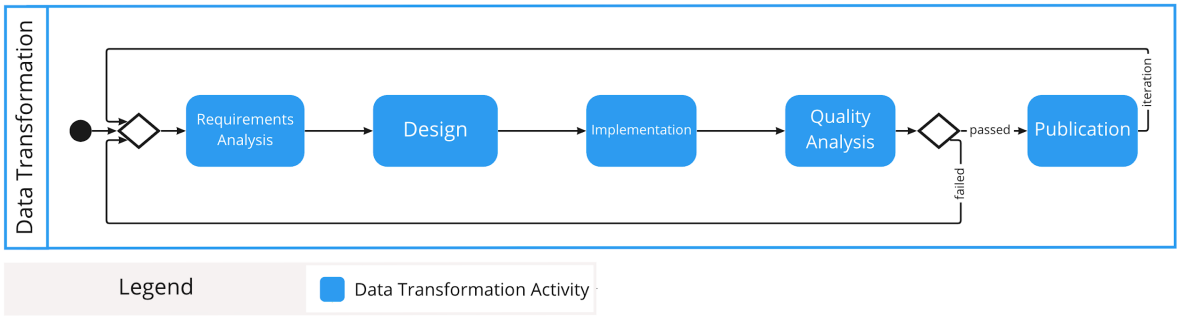

Figure 4: Generic Quality Management Process of Data Transformation: Visualization of the activities and their temporal dependencies in the generic Quality Management Process of Data Transformations. UML Activity Diagram, supplemented with blue activities for mapping to data transformations.

It is obvious that the three quality-focused data management processes considered and visualized separately are similar on an abstract level. The number of activities, their denominations and connections are the same at the level of abstraction shown above. The differences become visible with the detailed definitions of the activities. In a quality-focused data management process that distinguishes between data, data models and data transformations, it is important to define both the internal and the cross-cutting connections. Accordingly, *Figure 5* shows the integrated quality-focused data management process that represents the activities of data, data models, and data transformations at once and defines their intra- and interconnections. The green color highlights the activities of data, the orange color highlights the activities of data models, and the blue color highlights the activities of data transformations.

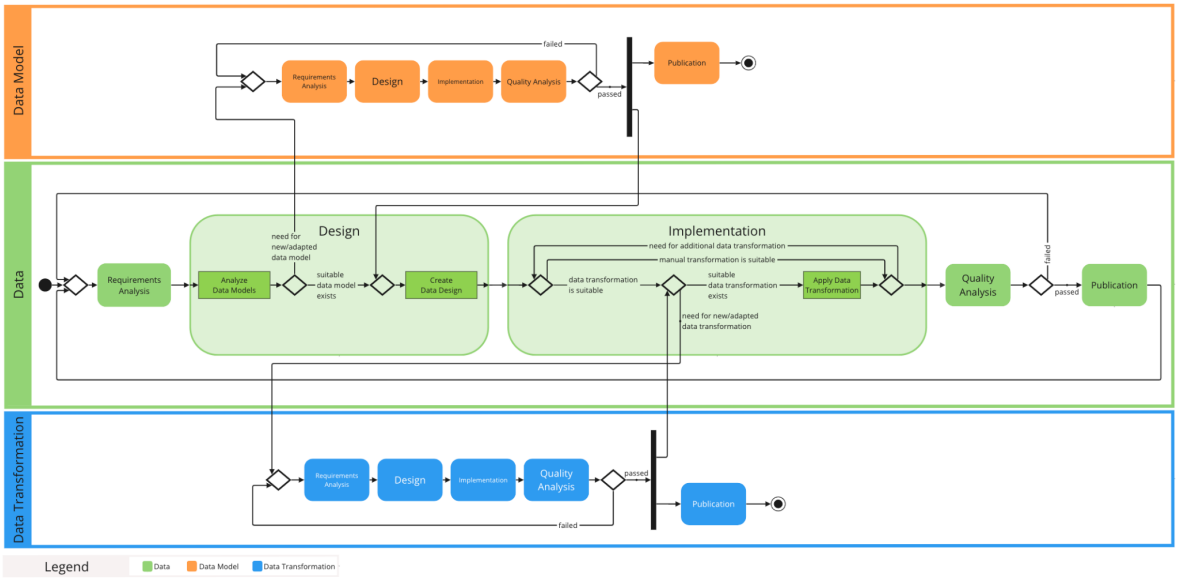

Figure 5: Generic Quality-Focused Data Management Process that distinguishes between Data, Data Model and Data Transformation: Visualization of the activities and their dependencies in the generic quality-focused data management processes for Data (green), Data Models (orange), and Data Transformations (blue). Data Design and Data Implementation are enlarged to illustrate within these activities decisions and processes like tasks, displayed as UML Actions. Formally, it is an UML Activity Diagram with additions for color mapping to data, data models, and data transformations.

A characteristic of the integrated process is the *concurrent activities*. Considering the separately defined activities for data, data models, and data transformations combined in an integrated process, not only dependencies but also concurrencies may arise. More precisely, this refers to the activities Data Design and Data Implementation. The data design activity can overlap with all of the data model activities (Data Model Requirements Analysis, Design, Implementation, and Quality Analysis), except *(Data Model) Publication* since this activity should be independent. This overlap is the case when it is decided to create or adapt a data model within the Data Design activity. Hence, by definition, these activities are concurrent. Data Implementation is the other concurrent example of this process. The Data Transformation activities Requirements Analysis, Design, Implementation, and Quality Analysis are concurrent to this activity if there is a need for a new or adapted data transformation.
Figure 5 shows the temporal representation from a data perspective. Therefore, the starting point is in the swimlane of data and data model activities and data transformation activities should be gone through only in case of new implementations. Another characteristic of this integrated process is the closer look at the two concurrent activities Data Design and Data Implementation. To illustrate the decisions and processes within these two concurrent activities, they are enlarged in *Figure* 5. The magnification includes decisions within activities and tasks represented as UML Actions. Thus, after deciding which data transformation is useful, it should be applied (cf. Action *Apply Data Transformation*). During the Data Design activity, the decision has to be made whether to use an existing data model or to create a new one. Analogously, during the Data Transformation activity the decision has to be made between an existing data transformation or a new data transformation. Additionally, it should be decided whether a manual implementation is useful or even a combination between manual implementation and transformation by implementation. This apparent difference is illustrated in the following chapters by defining *Data Transformation* as a UML ActivityGroup, grouping two activities for manual and automated transformation.

Causal Representation

The causal visualization is similar to the UML 2.2 specification. As before, activity nodes represent activities but we do not model an activity flow. Instead, only dependencies are modeled in the form of object flows. Here, large square objectNodes represent the artifacts that are attached as outputs to the respective activity. To minimize the representation and increase clarity, the input artifacts are modeled only by the directed arrows (objectFlow) of the object nodes.

To identify the causal dependencies between activities of data, data models, and data transformations, it is useful to model the dependencies for data, data models, and data transformations separately. *Figure 6* therefore, gives an overview of the causal dependencies in a quality-focused data management process. The representation is reduced to artifacts that have causal dependencies within data activities. Artifacts from activities that are not shown and dependencies that have arisen through iterations are not shown as well. For this visualization, the Data Implementation activity is shown as an UML *ActivityGroup*, showing differences for the dependencies diagram in *Figure 9*. The details of the activity are shown in the detailed definition of the activities.

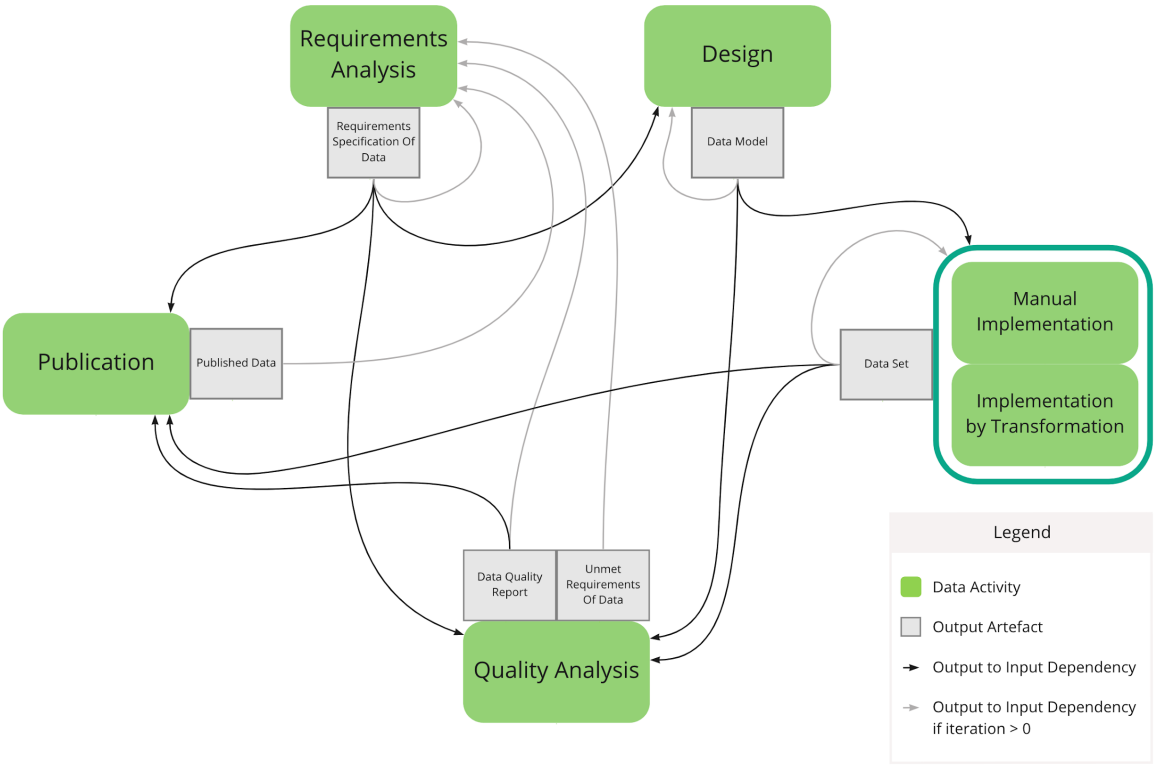

Figure 6: Generic Quality-Focused Data Management Process for Data: Visualization of the causal dependencies in the generic quality-focused management process for data via a dependency diagram that is inspired by UML activity diagrams. Dependencies are modeled in the form of object flows. Here, objectNodes represent the artifacts that are attached as outputs to the respective activity. To minimize the representation and increase clarity, the input artifacts are modeled only by directed arrows (objectFlow) from the object nodes. The activities *Manual Implementation* and *Implementation by Transformation* are grouped, because they have the same input and output artifacts in that visualization.

Analogous to the causal dependencies of data, the causal dependencies of data models are also modeled separately. *Figure 7* presents the data model activities and the internal causal dependencies and *Figure 8* presents the data transformation activities and the internal causal dependencies.

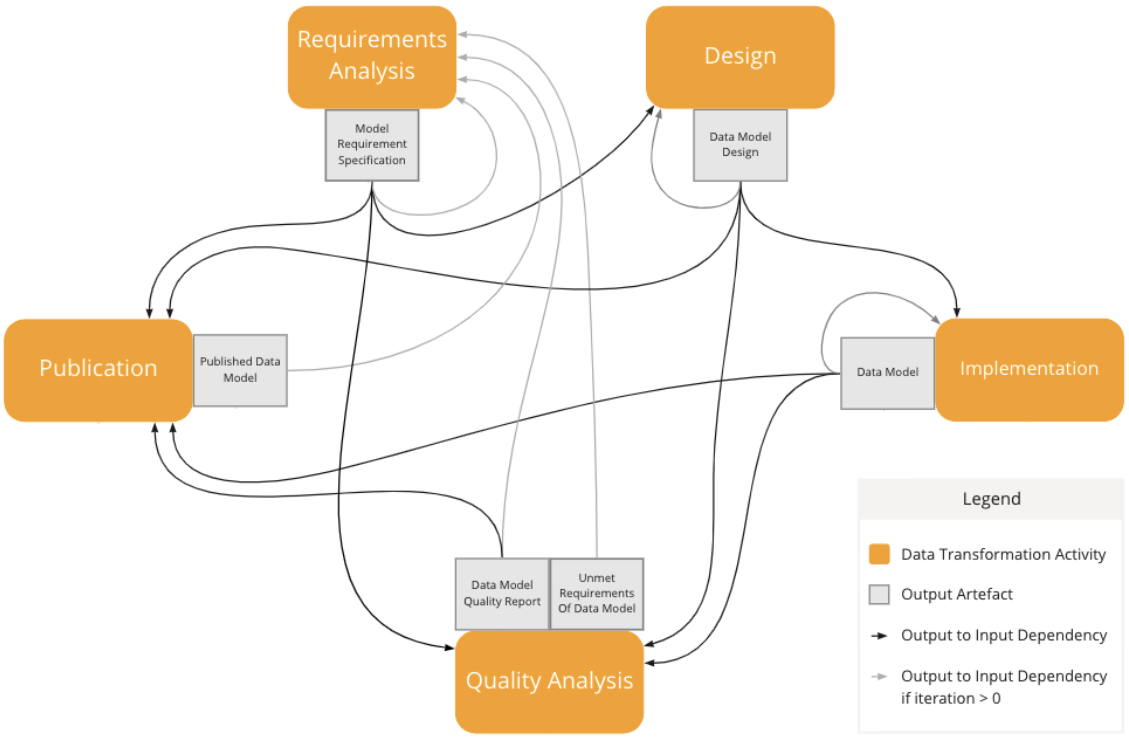

Figure 7: Generic Quality-Focused Data Management Process for Data Model: Visualization of the causal dependencies in the generic quality-focused data management process for data models via a Dependency diagram that is inspired by UML activity diagrams. Dependencies are modeled in the form of object flows. Here, objectNodes represent the artifacts that are attached as outputs to the respective activity. To minimize the representation and increase clarity, the input artifacts are modeled only by directed arrows (objectFlow) from the object nodes.

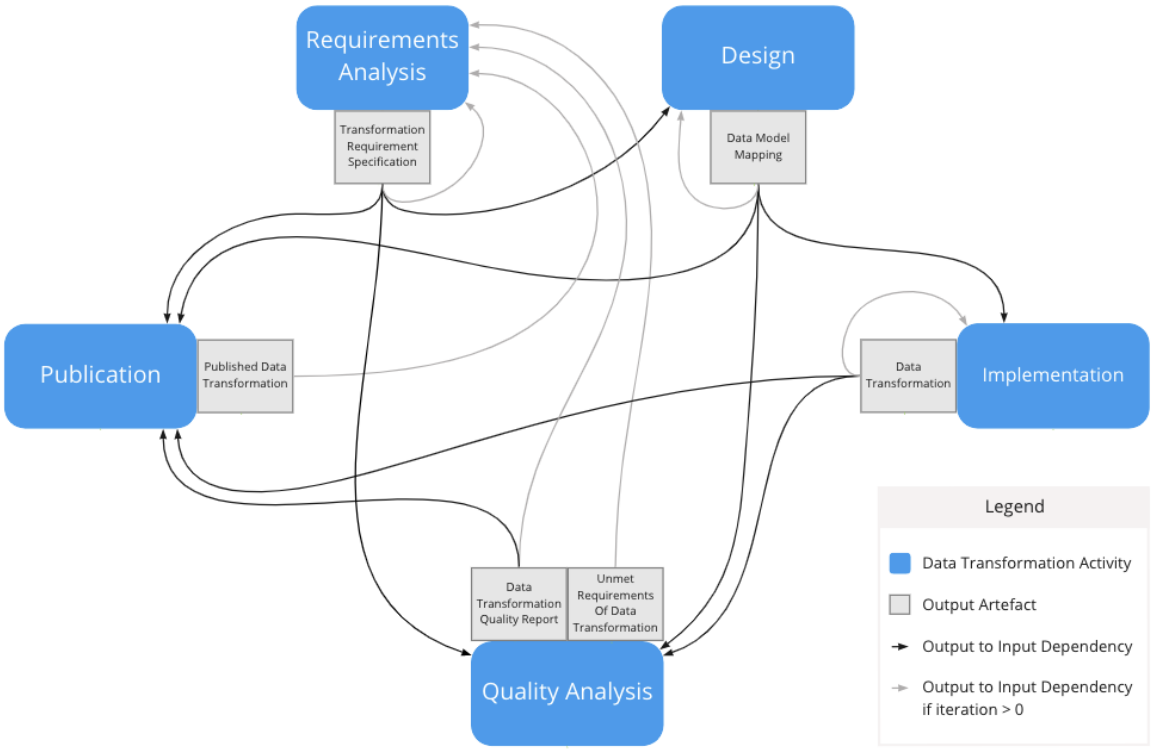

Figure 8: Generic Quality-Focused Data Management Process for Data Transformation: Visualization of the causal dependencies in the generic quality-focused data management process for data transformations via a Dependency diagram that is inspired by UML activity diagrams. Dependencies are modeled in the form of object flows. Here, objectNodes represent the artifacts that are attached as outputs to the respective activity. To minimize the representation and increase clarity, the input artifacts are modeled only by directed arrows (objectFlow) from the object nodes.

The three representations of causal dependencies that are considered and visualized separately are similar on an abstract level. The number of dependencies and their directions are the same at the level of abstraction shown above. The differences become visible with the detailed definitions of the activities.

In a quality-focused data management process that distinguishes between data, data models and data transformations, it is important to define both the internal and the cross-cutting dependencies. Accordingly, *Figure 9* shows the dependencies of the integrated quality-focused data management process that represents the activities of data, data models, and data transformations at once and defines their intra-dependencies. Here, as before, the green color highlights the activities of data, the orange color highlights the activities of data models, and the blue color highlights the activities of data transformations. Dependencies within data, data models and data transformations are not shown in this representation.

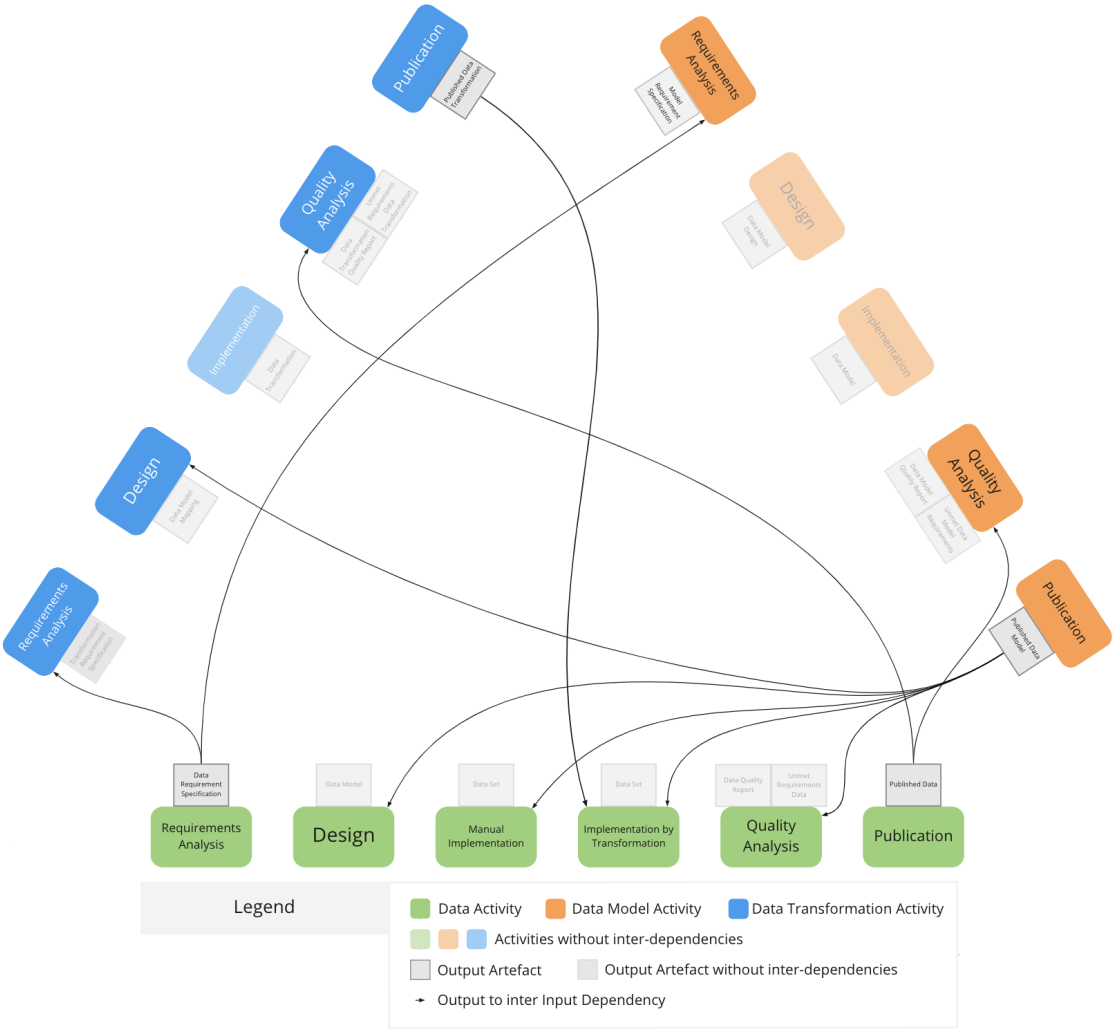

Figure 9: Generic Quality-Focused Data Management that distinguishes between Data, Data Model and Data Transformation: Visualization of the causal dependencies in the generic quality-focused data management process between data, data models and data transformations. Dependencies within data, data models and data transformations are outsourced to Figures 6, 7 and 8. This diagram is a dependency diagram that is inspired by UML activity diagrams. Dependencies are modeled in the form of object flows. Here, objectNodes represent the artifacts that are attached as outputs to the respective activity. To minimize the representation and increase clarity, the input artifacts are modeled only by directed arrows (objectFlow) from the object nodes.

Activities

The process describes activities that interact and interrelate with other activities. Activities are assigned either to the scope of data, data models, or data transformations. Therefore, activities identify themselves by the prefix of the scope and the name of the activity. The definition of each activity follows a uniform structure.

[*scope*][*name of activity*]

- **Input**: artifacts that should be present as a condition for the activity to be fulfilled
- **Output**: artifacts that are created as a result of the activity
- **Description**: a mainly informative form of presentation that describes the activity in a fact-based manner

- **ConstructiveQATasks**: tasks that are necessary to ensure appropriate quality of the output during the activity.
- **AnalyticQATasks**: tasks that are necessary to ensure appropriate quality of the output on the completed artifact
- **ConNextActivity/conStartActivity/conEndActivity/hasConcurrentActivity**: An activity indicates connections with other activities to show dependencies. Here, such activities are explicitly mentioned, which are explained below.

The activities are linked to one or more other activities via the "conNextActivity" class. An activity may be concurrent to multiple activities (via the “hasConcurrentActivity” class) that. These activities are called concurrentActivities.

Since activities convert inputs into outputs, the goal of any activity is to produce the output of optimal quality. To achieve this goal, constructive and/or analytical quality assurance tasks are assigned to each activity to assure the quality of the output. To ensure this, each task includes appropriate techniques.

This section contains the described activities grouped by Data, Data Models, and Data Transformations. Within the groups, these activities are sorted by the order in the process. The green color of the prefixed scope in brackets highlights the activities of Data to differentiate activities with the same name (e.g., Design). Analogously, the orange color highlights the activities of data models and the blue color highlights the activities of data transformations.

Data Activities

(Data) Requirement Analysis

- **Input**: Ar_UnmetRequirementsOfData (if iteration > 0); Ar_DataQualityReport (if iteration > 0); Ar_PublishedDataset (if iteration > 0); Ar_RequirementSpecificationOfData (if iteration > 0)
- **Output**: Ar_RequirementSpecificationOfData
- **Description**: The content of the data is defined. This includes which data will be collected and which methods should be used. Derive new or modify existing specific requirements of the data based on the purpose of the targeted data and (if available) usage problems, quality problems, and unmet requirements. This usually includes a current state, target state, and functional and non-functional requirements. Unmet quality problems, existing problems from use, or unmet requirements, should be analyzed first if they exist.
- **ConstructiveQATasks**: Ta_define_requirements_specification
- **AnalyticQATasks**: Ta_analyze_requirements_specification
- **conNextActivity**: (Data) Design (should)

(Data) Design (concurrent)

- **Input**: Ar_RequirementSpecificationOfData; Ar_PublishedDataModel (if iteration > 0); Ar_DataDesign (if iteration > 0); Ar_DataModel (if iteration > 0)

- **Output**: Ar_DataModel; Ar_DataDesign
- **Description**: The structure of data is defined by the form of the data model and the data structure. Accordingly, this activity includes gaining knowledge of data models that are relevant according to the Data Requirements Specification. The analysis should reveal whether to edit an existing data model, create a new data model, or choose an existing data model. For creating new data models, the data model activities should be concurrent with this activity.
- **ConstructiveQATasks**: Ta_create_data_design
- **AnalyticQATasks**: Ta_analyze_existing_data_models
- **hasConcurrentActivity**: (Data Model) Requirements Analysis; (Data Model) Design; (Data Model) Implementation; (Data Model) Quality Analysis
- **conNextActivity**: (Data) Manual Implementation (or); (Data) Implementation by Transformation (or)

(Data) Implementation (activityGroup)

- **Input**: *all artifacts of* (Data) Manual Implementation *and* (Data) Implementation by Transformation
- **Output**: *all artifacts of* (Data) Manual Implementation *and* (Data) Implementation by Transformation
- **Description**: This activity contains the manual implementation and the implementation by transformation. Both activities can be done multiple times and in combination. Here, a decision is made about whether a manual or automated transformation makes sense and whether multiple transformations are needed.
- **ConstructiveQATasks**: *all tasks of* (Data) Manual Implementation *and* (Data) Implementation by Transformation
- **AnalyticQATasks**: *all tasks of* (Data) Manual Implementation *and* (Data) Implementation by Transformation
- **conNextActivity**: *all activities of* (Data) Manual Implementation *and* (Data) Implementation by Transformation
- **consists of**: (Data) Manual Implementation; (Data) Implementation by Transformation

(Data) Manual Implementation

- **Input**: Ar_DataModel; Ar_DataSet (if iteration > 0); Ar_DataDesign
- **Output**: Ar_DataSet
- **Description**: This activity contains the manual creation and modification of data. Manual implementation includes manual collection and modification (including quality improvement) of data to one more data set(s).
- **ConstructiveQATasks**: Ta_perform_data_quality_improvement;
- **AnalyticQATasks**: Ta_analyze_data_model_validity; Ta_verify_data_conformance;
- **conNextActivity**: (Data) Quality Analysis (should)

(Data) Implementation by Transformation (concurrent)

- **Input**: Ar_DataModel; Ar_PublishedDataTransformation; Ar_DataSet; Ar_DataDesign
- **Output**: Ar_DataSet
- **Description**: This activity contains all modifications of data that is based on Data Transformations. Consequently, this activity contains all the activities of Data

Transformation except Publication. The activity (Data Transformation) Requirements Analysis should be the first one. This activity should be preceded by gaining knowledge of data transformations that are relevant according to the Data Transformation Requirements Specification. The analysis should reveal whether an existing data transformation can be used, adapted or a new data transformation created. For creating new data transformations, the data transformation activities should be concurrent with this activity. The final part of this activity is the application of the data transformation.

- **ConstructiveQATasks**: *all ConstructiveQATasks of concurrent activities*
- **AnalyticQATasks**: *all AnalyticQATasks of concurrent activities*
- **hasConcurrentActivity**: (Data Transformation) Requirements Analysis; (Data Transformation) Design; (Data Transformation) Implementation; (Data Transformation) Quality Analysis; (Data Transformation) Publication
- **conNextActivity**: (Data) Quality Analysis (should)

(Data) Quality Analysis

- **Input**: Ar_DataSet; Ar_RequirementSpecificationOfData; Ar_PublishedDataModel
- **Output**: Ar_DataQualityReport; Ar_UnMetRequirementsOfData; Ar_DataQualityReport
- **Description**: This activity contains the analysis of the data regarding specific quality aspects and dimensions. This can be done manually or with specific analysis techniques. The goal is to check whether all requirements are met. If problems arise, (Data) Requirements Analysis should be the next activity. If there are no problems or the problems are negligible, the (Data) Publication should be next.
- **ConstructiveQATasks**: Ta_create_quality_analysis_plan; Ta_create_data_quality_report;
- **AnalyticQATasks**: Ta_perform_data_quality_analysis; Ta_check_requirements;
- **conNextActivity**: (Data) Requirement Analysis (should); (Data) Publication (should)

(Data) Publication

- **Input**: Ar_DataSet; Ar_RequirementSpecificationOfData; Ar_DataQualityReport
- **Output**: Ar_PublishedDataset
- **Description**: This activity includes making the data publicly accessible, citable and identifiable. This includes research publications such as journals, publication of data in public repositories or data centers and activities that promote transparency and reproducibility of data and results.
- **ConstructiveQATasks**: Ta_release_data;
- **AnalyticQATasks**: Ta_verify_availability; Ta_verify_rights
- **conNextActivity**: (Data) Requirements Analysis (should)

Data Model Activities

(Data Model) Requirements Analysis (concurrent)

- **Input**: Ar_RequirementSpecificationOfData; Ar_DataModelQualityReport (if iteration > 0); Ar_PublishedDataModel (if iteration > 0); Ar_UnmetRequirementsOfDataModel (if iteration > 0); Ar_RequirementSpecificationOfDataModel (if iteration > 0)
- **Output**: Ar_RequirementSpecificationOfDataModel

- **Description**: Derive new or modify existing specific requirements of the data model based on the data requirement specification. This usually includes a current state, target state, as well as functional and non-functional requirements. Unmet quality problems, existing problems from use, or unmet requirements, should be analyzed first if they exist. This activity may be concurrent to (Data) Design if a new data model is created dedicatedly as part of a Data Design.
- **ConstructiveQATasks**: Ta_define_requirements_specification
- **AnalyticQATasks**: Ta_analyze_requirements_specification
- **hasConcurrentActivity**: (Data) Design
- **conNextActivity**: (Data Model) Design (should)

(Data Model) Design (concurrent)

- **Input**: Ar_RequirementSpecificationOfDataModel; Ar_DataModelDesign (if iteration > 0)
- **Output**: Ar_DataModelDesign
- **Description**: Create or modify the data model design. A modification of the data model may be necessary if the requirements have changed, quality problems have arisen or problems have arisen in the use of the data model. This activity may be concurrent to (Data) Design if a new data model is created dedicatedly as part of a Data Design.
- **ConstructiveQATasks**: Ta_design_data_model
- **AnalyticQATasks**: Ta_verify_data_model_design
- **hasConcurrentActivity**: (Data) Design
- **conNextActivity**: (Data Model) Implementation (should)

(Data Model) Implementation (concurrent)

- **Input**: Ar_DataModelDesign; Ar_DataModel (if iteration > 0)
- **Output**: Ar_DataModel
- **Description**: Implement a new data model or change the implementation of an existing data model within the technical environment based on the data model design. This activity may be concurrent to (Data) Design if a new data model is created dedicatedly as part of a Data Design.
- **ConstructiveQATasks**: Ta_implement_data_model; Ta_create_data_model_documentation; Ta_perform_data_model_quality_improvement (if iteration > 0)
- **AnalyticQATasks**: Ta_verify_data_model_quality;
- **hasConcurrentActivity**: (Data) Design
- **conNextActivity**: (Data Model) Quality Analysis (should)

(Data Model) Quality Analysis (concurrent)

- **Input**: Ar_DataModel; Ar_DataModelDesign; Ar_RequirementSpecificationOfDataModel;
- **Output**: Ar_DataModelQualityReport; Ar_UnmetRequirementsOfDataModel
- **Description**: Analysis of the data model regarding specific quality aspects and dimensions. Check whether requirements are met. If problems arise, (Data Model) Requirements Analysis should be next. If there are no problems or the problems are negligible, the (Data Model) Publication should be next. This activity may be

concurrent to (Data) Design if a new data model is created dedicatedly as part of a Data Design.
- **ConstructiveQATasks**: Ta_create_quality_analysis_plan
- **AnalyticQATasks**: Ta_perform_data_model_quality_analysis; Ta_check_requirements; Ta_perform_sample_validation;
- **hasConcurrentActivity**: (Data) Design
- **conNextActivity**: (Data Model) Publication (should); (Data Model) Requirement Analysis (should)

(Data Model) Publication

- **Input**: Ar_DataModel; Ar_DataModelQualityReport
- **Output**: Ar_PublishedDataModel
- **Description**: Making the data model publicly accessible, citable, identifiable and reusable.
- **ConstructiveQATasks**: Ta_release_data_model
- **AnalyticQATasks**: Ta_verify_availability; Ta_verify_rights
- **conNextActivity**: (Data Model) Requirements Analysis (should)

Data Transformation Activities

(Data Transformation) Requirement Analysis (concurrent)

- **Input**: Ar_PublishedDataModel; Ar_RequirementSpecificationOfData; Ar_DataTransformationQualityReport (if iteration>0); Ar_UnmetRequirementsOfDataTransformation (if iteration>0); Ar_RequirementSpecificationOfDataTransformation (if iteration > 0)
- **Output**: Ar_RequirementSpecificationOfDataTransformation
- **Description**: Derive new or modify existing specific requirements of the data model based on the data requirement specification. This usually includes a current state, target state, as well as functional and non-functional requirements. Unmet quality problems, existing problems from use, or unmet requirements, should be analyzed first if they exist. This activity may be concurrent to (Data) Implementation by Transformation if a new data transformation is created dedicatedly as part of (Data) Implementation by Transformation.
- **ConstructiveQATasks**: Ta_define_requirements_specification
- **AnalyticQATasks**: Ta_analyze_requirements_specification
- **hasConcurrentActivity**: (Data) Implementation by Transformation
- **conNextActivity**: (Data Transformation) Design (should)

(Data Transformation) Design (concurrent)

- **Input**: Ar_PublishedDataModel; Ar_RequirementSpecificationOfDataTransformation
- **Output**: Ar_DataModelMapping
- **Description**: Conceptual design of a data transformation e.g. in the form of a mapping of one source and one target data model. For the implementation, it is not important to have the data model itself as an input, but the technical information of the two data models. This information should be part of the data model mapping. This activity may be concurrent to (Data) Implementation by Transformation if a new data

transformation is created dedicatedly as part of (Data) Implementation by Transformation.

- **ConstructiveQATasks**: Ta_create_data_model_mapping
- **AnalyticQATasks**: Ta_verify_data_transformation_design
- **hasConcurrentActivity**: (Data) Implementation by Transformation
- **conNextActivity**: (Data Transformation) Implementation (should)

(Data Transformation) Implementation (concurrent)

- **Input**: Ar_DataModelMapping
- **Output**: Ar_DataTransformation
- **Description**: Implementation of the data transformation. This can include automatic implementation by the tool when using tools for data transformations and a corresponding data mapping. This activity may be concurrent to (Data) Implementation by Transformation if a new data transformation is created dedicatedly as part of (Data) Implementation by Transformation.
- **ConstructiveQATasks**: Ta_implement_data_transformation; Ta_perform_data_transformation_quality_improvement (if iteration > 0);
- **AnalyticQATasks**: Ta_verify_data_transformation_quality;
- **hasConcurrentActivity**: (Data) Implementation by Transformation
- **conNextActivity**: (Data Transformation) Quality Analysis (should)

(Data Transformation) Quality Analysis (concurrent)

- **Input**: Ar_DataModelMapping; Ar_PublishedDataset; Ar_RequirementSpecificationOfDataTransformation; Ar_DataTransformation
- **Output**: Ar_DataTransformationQualityReport; Ar_UnmetRequirementsOfDataTransformation
- **Description**: Analysis of the data transformation regarding specific quality aspects and dimensions. Check whether requirements are met. If problems arise, (Data Transformation) Requirements Analysis should be next. If there are no problems or the problems are negligible, the (Data Transformation) Publication should be next. This activity may be concurrent to (Data) Implementation by Transformation if a new data transformation is created dedicatedly as part of (Data) Implementation by Transformation.
- **ConstructiveQATasks**: Ta_create_quality_analysis_plan
- **AnalyticQATasks**: Ta_perform_data_transformation_quality_analysis; Ta_check_requirements; Ta_perform_data_transformation_sample
- **hasConcurrentActivity**: (Data) Implementation by Transformation
- **conNextActivity**: (Data Transformation) Publication (or); (Data Transformation) Requirement Analysis (or)

(Data Transformation) Publication

- **Input**: Ar_DataModelMapping; Ar_DataTransformation; Ar_DataTransformationQualityReport
- **Output**: Ar_PublishedDataTransformation
- **Description**: Making the data model mapping and the data transformation publicly accessible, citable, identifiable and reusable.
- **ConstructiveQATasks**: Ta_release_data_transformation

- **AnalyticQATasks**: Ta_verify_availability; Ta_verify_rights
- **conNextActivity**: (Data Transformation) Requirements Analysis (should)

Tasks

The process describes tasks as part of activities. For this, the tasks relate to quality assurance, not data management. The tasks are either constructive or analytical and are identified by the prefix "Ta_" and the name of the task. All tasks are defined using a uniform structure, which is defined as follows:

Ta_[*name of task*]

- **Description**: A mainly informative form of presentation that describes the task in a fact-based manner
- **Skills**: An ability to perform this task
- **Tool**: Tool-based techniques that support users by performing this task
- **Utility**: Utility-based techniques that support users by performing this task
- **Guidance**: Guidance-based techniques that support users by performing this task

Each task is defined and described separately and is linked via the activities. In the "Tasks" section, all tasks are listed in alphabetical order. The following list of tasks is sorted by alphabetical order.

Ta_analyze_data_model_validity

- **Description**: A data model should have been chosen or created before implementing data. Ideally, the creation of data is accompanied by constant validation. Nevertheless, the user should perform a subsequent validation after manual data implementation. The validation itself can be realized using tools for acquisition with built-in validation, editors that enable attending validation, and human matching of data acquisition policies.
- **Skills**: Sk_analzye_compliance; Sk_structural_thinking
- **Tool**: Te_validate_data_by_tool
- **Utility**: Te_validate_data_by_utility
- **Guidance**: Te_validate_data_by_guidance

Ta_analyze_existing_data_models

- **Description**: Often, a variety of existing data models may be appropriate for data design. Thus, it should be thoroughly considered whether an existing data model is suitable, can be adapted, or a new data model should be created.
- **Skills**: Sk_analytic_thinking; Sk_data_modelling; Sk_data_model_design
- **Tool**: -
- **Utility**: -
- **Guidance**: -

Ta_analyze_requirement_specification

- Description: The requirement specification should be based on IEEE Software Requirements Specification (SRS). An SRS should be a) Correct; b) Unambiguous;

c) Complete; d) Consistent; e) Ranked for importance and/or stability; f) Verifiable; g) Modifiable; h) Traceable (Institute of Electrical and Electronics Engineers (1998) IEEE recommended practice for software requirements specifications: approved 25 June 1998, IEEE-SA Standards Board). This so-called requirements review needs to be checked manually or by tool support. “A requirements review is a structured process where key stakeholders from the user groups and the project team walk through the requirements document line-by-line. They analyze the requirements looking for problems to ensure the requirements are complete, correct, clear, and represent an accurate and mutual understanding among all of the stakeholders.”

- **Skills**: Sk_general_domain_knowledge; Sk_analytic_thinking; Sk_define_requirements
- **Tool**: Te_requirements_review_by_tool;
- **Utility**: -
- **Guidance**: Te_requirements_review_by_guidance;

Ta_check_requirements

- **Description**: With the requirement specification as an input, a check is to be performed whether each requirement is fulfilled or not. This includes the implementation of checks and their execution.
- **Skills**: Sk_analytical_thinking; Sk_general_domain_knowledge
- **Tool**: Te_check_requirements_by_tool
- **Utility**: -
- **Guidance**: Te_check_requirements_by_guidance

Ta_create_data_design

- Description: A data design illustrates how data should be structured and used in the lifecycle, their relationships, and how it can be organized. An essential component is the data model that facilitates a deeper understanding of the stored data. Furthermore, it is important to define rules and methods for how the data should be generated and processed.
- **Skills**: Sk_structural_thinking; Sk_data_design
- **Tool**: -
- **Utility**: -
- **Guidance**: -

Ta_create_data_model_documentation

- **Description**: Use common documentation languages to create a technology-dependent data model documentation. The documentation is a crucial part of the data model.
- **Skills**: Sk_specific_data_model_knowledge; Sk_data_model_design; Sk_documentation
- **Tool**: Te_create_documentation_by_tools
- **Utility**: Te_create_documentation_by_generators
- **Guidance**: Te_create_documentation_by_documentation_language

Ta_create_quality_analysis_plan

- **Description**: Create a quality analysis plan that contains how the quality analysis should be done and which techniques should be used. This depends very much on the type of data (/data model/data transformation) and the requirements available. There are guidelines to help users create a plan. However, these are either context-specific or very general.
- **Skills**: Sk_data_management; Sk_analytic_thinking; Sk_structural_thinking
- **Tool**: -
- **Utility**: -
- **Guidance**: Te_create_quality_analysis_plan_by_guidance

Ta_create_data_quality_report

- **Description**: A data quality report should be the result of a data quality analysis that contains the exploration and verification of data. Depending on the kind of quality analysis, the report may contain different aspects. There are quality analysis tools that create such quality reports as a result (e.g. data quality KPIs). There are also best practices or documentation that guide the users with questions to create a quality report manually.
- **Skills**: Sk_structural_thinking; Sk_documentation; Sk_analyze_quality
- **Tool**: Te_create_data_quality_report_by_tool
- **Utility**: -
- **Guidance**: Te_create_data_quality_report_by_guidance

Ta_create_data_model_mapping

- **Description**: Data (model) mapping is the process of creating a mapping of all data elements between two distinct data models. (Wikipedia) Data mapping can be implemented algorithmically in various ways. These include, for example, implementation using procedural code, the use of XSLT transformations, or using graphical mapping tools that automatically create executable transformation programs. (Wikipedia)
- **Skills**: Sk_specific_data_model_knowledge; Sk_use_of_data_mapping_tool; Sk_analytical_thinking
- **Tool**: Te_data_model_mapping_by_tool
- **Utility**: -
- **Guidance**: -

Ta_define_requirements_specification

- **Description**: The requirement specification should comply with standard specification techniques, such as IEEE 830-1998 or IEEE 1233-1998. There are existing tools and guidelines to support users in retaining these standards.
- **Skills**: Sk_general_domain_knowledge; Sk_analytic_thinking; Sk_define_requirements
- **Tool**: Te_requirements_specification_by_tool;
- **Utility**: -

- **Guidance**: Te_use_requirements_specification_technique; Te_use_requirements_specification_templates;

Ta_design_data_model

- **Description**: A data model design defines a concept on the restrictions of the data model. (Wikipedia) Therefore, the design should stick with standard data model design languages such as Entity Relationship Model (ERM) or Unified Modeling Language (UML). Both tools and guidelines exist to help users create them. If an existing model is to be further developed or restricted, workflow specifications usually exist to ensure the quality of this process.
- **Skills**: Sk_structural_thinking; Sk_data_modelling; Sk_analytic_thinking;
- **Tool**: Te_design_data_model_by_tool
- **Utility**: -
- **Guidance**: Te_design_data_model_by_design_language; Te_design_data_model_by_workflow

Ta_implement_data_model

- **Description**: A data model is of high quality if it implements the data model according to the data model design and meets the requirements according to the requirements specification. Tools exist that automatically generate an implementation of the data model based on the UML or ER diagram of the data model.
- **Skills**: Sk_structural_thinking; Sk_analytic_thinking; Sk_programming
- **Tool**: Te_implement_data_model_by_tool
- **Utility**: -
- **Guidance**: Te_implement_data_model_by_workflow; Te_use_code_project_qa

Ta_implement_data_transformation

- **Description**: Data mapping can be implemented algorithmically in various ways. These include, for example, implementation using procedural code, the use of XSLT transformations, or using graphical mapping tools that automatically create executable transformation programs. (Wikipedia)
- **Skills**: Sk_programming; Sk_use_of_data_mapping_tool; Sk_analytic_thinking
- **Tool**: Te_data_model_mapping_by_tool; Te_data_transformation_generation_by_tool;
- **Utility**: -
- **Guidance**: Te_create_tests_first_approach;

Ta_perform_data_model_quality_analysis

- **Description**: The implemented data model needs to be checked for quality problems. This can be done exploratively and detective.
- **Skills**: Sk_analytical_thinking; Sk_general_domain_knowledge
- **Tool**: Te_perform_explorative_data_quality_analysis; Te_perform_detective_data_quality_analysis
- **Utility**: -
- **Guidance**: Te_perform_detective_data_quality_analysis_by_guidance

Ta_perform_data_model_quality_improvement

- **Description**: Fix problems that relate to previous versions of the same data model. This depends on the technology, the quality tool, and the used technique to implement the data model.
- **Skills**: Sk_analytical_thinking; Sk_general_domain_knowledge
- **Tool**: Te_perform_data_model_quality_improvement_by_tool
- **Utility**: -
- **Guidance**: Te_perform_data_model_quality_improvement_by_guidance

Ta_perform_data_quality_analysis

- **Description**: The implemented data needs to be checked for quality problems. This should be done explorativelly and detective.
- **Skills**: Sk_analytical_thinking; Sk_general_domain_knowledge
- **Tool**: Te_perform_explorative_data_quality_analysis; Te_perform_detective_data_quality_analysis
- **Utility**: -
- **Guidance**: Te_perform_detective_data_quality_analysis_by_guidance

Ta_perform_data_quality_improvement

- **Description**: Based on the list of quality problems, improvement should take place in this task. This depends on the technology, the quality tool, and the used technique.
- **Skills**: Sk_analytical_thinking; Sk_general_domain_knowledge
- **Tool**: Te_perform_data_quality_improvement_by_tool
- **Utility**: Te_data_normalization_by_tool
- **Guidance**: Te_perform_data_quality_improvement_by_guidance

Ta_perform_data_transformation_quality_analysis

- **Description**: The implemented data transformation needs to be checked for quality problems. This should be done based on known code quality analysis techniques and automated software checks like vulnerability checks or language conformance checks.
- **Skills**: Sk_analytical_thinking; Sk_general_domain_knowledge
- **Tool**: Te_perform_data_transformation_quality_analysis_by_tool
- **Utility**: Te_perform_software_vulnerability_checks; Te_perform_software_language_conformance_checks
- **Guidance**: Te_data_transformation_quality_analysis_by_guidance

Ta_perform_data_transformation_quality_improvement

- **Description**: Based on the list of quality problems and known code improvement techniques, improvement should take place in this task. This depends on the technology, the quality tool, and the used technique.
- **Skills**: Sk_analytical_thinking; Sk_general_domain_knowledge
- **Tool**: Te_perform_code_improvement_by_tool
- **Utility**: Te_code_reviews; Te_perform_code_improvement_by_utility

- **Guidance**: Te_perform_code_improvement_by_guidance

Ta_perform_data_transformation_sample

- **Description**: A data transformation should finally be applied to a sample data set. Therefore, it is necessary to choose a source sample data set, perform the transformation and validate the target data set.
- **Skills**: Sk_analytical_thinking; Sk_general_domain_knowledge
- **Tool**: -
- **Utility**: Te_data_transformation_sample_testing_by_tool;
- **Guidance**: -

Ta_perform_sample_validation

- **Description**: Validate an existing sample data set with the implemented data model (e.g. schema file).
- **Skills**: Sk_execute_script
- **Tools**: -
- **Utility**: Te_perform_sample_validation_by_utility
- **Guidance**: -

Ta_release_data

- **Description**: The release of data can be done manually and semi-automatically on a website or a repository. To ensure the publication qualitatively (persistent identifier, citable, accessible), it is recommended to use publication tools, repositories or guidances. In order to be able to publish data, it is sometimes necessary to anonymize the data (e.g. medical data). Anonymization tools are recommended for this purpose.
- **Skills**: Sk_general_domain_knowledge
- **Tool**: Te_use_repositories_to_publish; Te_use_publication_tools; Te_anonymize_data_by_tool;
- **Utility**: -
- Guidance: Te_use_guidance_to_publish;

Ta_release_data_model

- **Description**: The release of the data model can be done manually and semi-automatically on a website or a repository. To ensure the publication qualitatively (persistent identifier, citable, accessible), it is recommended to use publication tools, repositories or guidances.
- **Skills**: Sk_general_domain_knowledge
- **Tool**: Te_use_repositories_to_publish; Te_use_publication_tools
- **Utility**: -
- **Guidance**: Te_release_data_model_by_guidance

Ta_release_data_transformation

- **Description**: The release of the data transformation can be done manually and semi-automatically on a website or a repository. To ensure the publication qualitatively (persistent identifier, citable, accessible), it is recommended to use publication tools, repositories or guidances.
- **Skills**: Sk_general_domain_knowledge
- **Tool**: Te_use_repositories_to_publish; Te_use_publication_tools
- **Utility**: -
- **Guidance**: Te_release_data_transformation_by_guidance

Ta_verify_availability

- **Description**: The permanent availability is important for the distribution of the work product. Therefore, monitoring of availability needs to be implemented and domain/community dependent networks have to be used.
- **Skills**: Sk_general_domain_knowledge;
- **Tool**: Te_check_publication_availability_by_tool
- **Utility**: Te_check_publication_availability_by_utility
- **Guidance**: Te_check_publication_domain_guidelines

Ta_verify_data_conformance

- **Description**: The data set(s) need to be checked in a consistent and systematic manner for language/technology, standards compliance and conventions (e.g. naming) according to the requirements specification and to the organization's guidelines/policies like acquisition guidelines to ensure various quality aspects such as traceability. Tools like editors may help organize data and support the user with built-in addons like linters.
- **Skills**: Sk_analzye_compliance; Sk_specific_domain_knowledge; Sk_structural_thinking
- **Tool**: Te_verify_data_conformance_by_validation_tool; Te_verify_data_language_conformance_by_tool
- **Utility**: Te_verify_data_conformance_by_validation_utility; Te_verify_data_language_conformance_by_utility
- **Guidance**: Te_verify_data_conformance_by_validation_guidance; Te_verify_data_language_conformance_by_guidance; Te_verify_data_requirements_by_guidance

Ta_verify_data_model_design

- **Description**: Checking the quality of data model designs is essential and impacts the quality of the data model. This includes verifying the data model design for compliance with modeling standards. This increases the quality of the data model design and thus impacts the quality of the data model. Depending on the modeling language, tools, utilities, and guidance support the user in the verification process.
- **Skills**: Sk_data_model_design; Sk_analzye_compliance
- **Tool**: Te_verify_data_model_design_quality_by_tool
- **Utility**: Te_verify_data_model_design_language_by_utility

- **Guidance**: Te_verify_data_model_design_language_by_guidance

Ta_verify_data_model_quality

- **Description**: The implemented data model and the related documentation need to be checked for language and standards compliance.
- **Skills**: Sk_analyze_quality; Sk_specific_data_model_knowledge; Sk_analytic_thinking
- **Tool**: -
- **Utility**: -
- **Guidance**: Te_verify_data_model_language; Te_verify_data_model_documentation

Ta_verify_data_transformation_design

- **Description**: Checking the quality of data transformation designs is essential and impacts the quality of the data model. This includes verifying the data transformation design for compliance with modeling standards. This increases the quality of the data transformation design and thus impacts the quality of the data transformation. Depending on the modeling language, tools, utilities, and guidance support the user in the verification process.
- **Skills**: Sk_data_transformation_design; Sk_use_of_data_mapping_tool; Sk_analzye_compliance
- **Tool**: Te_verify_data_transformation_design_quality_by_tool
- **Utility**: Te_verify_data_transformation_design_language_by_utility
- **Guidance**: Te_verify_data_transformationl_design_language_by_guidance

Ta_verify_data_transformation_quality

- **Description**: Once the implementation is done, verification is needed to ensure that the data transformation meets the quality standards set by the organization and it meets data transformation requirements set in advance. Since a data transformation is a software, the same methods apply here.
- **Skills**: Sk_analytic_thinking; Sk_structural_thinking; Sk_analyze_quality; Sk_programming
- **Tool**: Te_verify_software_quality_by_tool
- **Utility**: -
- **Guidance**: Te_verify_software_quality_by_guidance

Ta_verify_rights

- **Description**: Rights statements and license indications are crucial information for reuse. Furthermore, it's essential to verify rights and license information to standards.
- **Skills**: Sk_analzye_compliance
- **Tool**: -
- **Utility**: -
- **Guidance**: Te_verify_rights_by_guidance

Techniques

The process describes techniques as part of tasks. For this, the techniques relate to quality assurance, not data management.
The techniques are either tool-based, utility-based, or guidance-based and are identified by the prefix “Te_” and the name of the technique. All techniques are defined using a uniform structure, which is defined as follows:

Te_[*name of technique*]

- **Description**: A mainly informative form of presentation that describes the technique in a fact-based manner.
- **Tool/Utility/Guidance**: A formal term that describes the type of tool, utility, or guidance. Usually, examples are given there.

Each technique is defined and described separately and is linked via the tasks. The following list of techniques is sorted by alphabetical order.

Te_anonymize_data_by_tool

- **Description**: The use of tools that pseudonymizes data sets or delete personalized information (e.g. in case of privacy issues).
- **Tools**: Anonymization Tool (e.g. Amnesia)

Te_create_data_quality_report_by_guidance

- **Description**: Documents that guide the user in quality reporting. This is typically available as best practices.
- **Guidance**: Best Practices for Quality Reporting

Te_create_data_quality_report_by_tool

- **Description**: Tools that do quality analysis usually create a quality report as a result.
- **Tool**: Data Quality Reporting Tool (e.g. Gartner Magic Quadrant for Data Quality Solutions)

Te_create_documentation_by_documentation_language

- **Description**: A documentation should be created by the use of a standard documentation language. This is often technology dependent. For XSD schemas there are specific elements available.
- **Guidance**: Documentation Specification Language (e.g. xs:documentation)

Te_create_documentation_by_generators

- **Description**: “A documentation generator is a programming tool that generates software documentation intended for programmers (API documentation) or end users (end-user guide), or both, from a set of source code files, and in some cases, binary files.” (Wikipedia)
- **Utility**: Data Model Documentation Generator (e.g. Elasoft SqlSpec)

Te_create_documentation_by_tools

- **Description**: Tools that support users creating documentation. Depending on the technology, there are certain tools available that enable you to do e.g. describing tables and columns, other database objects, and generating convenient documents for sharing.
- **Tool**: Data Model Documentation Tool (e.g. Dataedo)

Te_create_tests_first_approach

- **Description**: Test-first is an approach to software design in test driven development in which the user creates unit tests based on specifications before writing the source code. Depending on the technology and the requirements this technique can reduce debugging time and improve working in a team.
- **Guidance**: Best Practices in Model Driven Development

Te_check_publication_availability_by_tool

- **Description**: Use Tools that monitor the availability of the work product.
- **Tool**: Monitoring Tool (e.g. Icinga)

Te_check_publication_availability_by_utility

- **Description**: Use scripts that monitor the availability of the work product.
- **Utility**: Availability Scripts (e.g. Cron Jobs)

Te_check_publication_domain_guidelines

- **Description**: Institutions or communities usually have guidelines or recommendations for publishing work products. The publication of the work should comply with these recommendations if they exist.
- **Guidance**: Publication Guidelines (e.g. NC State Repository Guidelines)

Te_create_quality_analysis_plan_by_guidance

- **Description**: Guidelines support the user to create a quality analysis plan. They are usually context-specific, very general or available as best practices.
- **Guidance**: Quality Analysis Plan Guidance (e.g. DataONE Best Practices)

Te_check_requirements_by_guidance

- **Description**: Check the requirements manually by the list of requirements
- **Guidance**: Requirements Specification Check Best Practices

Te_check_requirements_by_tool

- **Description**: Implement tests that check if requirements are met. Execute them
- **Tool**: Requirements Check Management Tools (e.g. Modern Requirements Management Tools)

Te_code_reviews

- **Description**: Asses any (especially new) code for bugs, errors, quality problems and vulnerabilities. There are many variations of code review processes. Depending on the technology there are tools that support the user in manual checking the code.
- **Tool**: Code Review Tool (e.g. Collaborator)

Te_data_model_mapping_by_tool

- **Description**: The use of graphical mapping tools that automatically create executable transformation programs. There are also tools available that support the user in the mapping itself. Semantic mapping is an approach that semantically maps element and attribute names automatically.
- **Tool**: Data Model Mapping Tool (e.g. Altova Map Force)

Te_data_normalization_by_tool

- **Description**: The use of tools that automatically normalize contents like measurements (e.g. cm/miles/dm to meter). Typically, these tools are components of larger tools and less standalone.
- **Tools**: Data Normalization Tool (e.g. HAWKSearch)

Te_data_transformation_generation_by_tool

- **Description**: Describes the automatic code generation that is based on the data model mapping that is done beforehand.
- **Tool**: Data Transformation Generator by Data Mapping (e.g. IBM InfoSphere)

Te_data_transformation_sample_testing_by_tool

- **Description**: Use the transformation script or tool to create data set(s) based on sample data set(s). Thrown errors are a clear sign of faulty transformations
- **Tool**: Data Transformation Script/Tool (e.g. Oxygen with XSLT Script)

Te_design_data_model_by_workflow

- **Description**: Institutions and communities offer workflows to create data models for either existing common data models, for standardized data ingests or data with special requirements. These workflows often begin with a template file and create output files by scripts to ensure the quality of the output.
- **Guidance**: Data Model Template/Workflow File (e.g. LIDO Profile Workflow)

Te_design_data_model_by_design_language

- **Description**: Data Model designs are usually realized by Entity Relationship Model (ERM) or Unified Modeling Language (UML) that provide a standard way to visualize the design of a data model.
- **Guidance**: Data Model Design Language (e.g. UML)

Te_design_data_model_by_tool

- **Description**: Use of tools that support users to create Entity Relationship Model (ERM) or Unified Modeling Language (UML) models
- **Tool**: Data Design Modeling Tool (e.g. Oracle SQL Developer Data Modeler)

Te_implement_data_model_by_workflow

- **Description**: Use of workflows and frameworks to automatically create data models. Typically realized by Templates and/or Scripts.
- **Guidance**: Data Model Template File (e.g. LIDO Profile ODD Template)

Te_implement_data_model_by_tool

- **Description**: Typically realized by tools that create data models based on Entity Relationship Model (ERM) or Unified Modeling Language (UML) models.
- **Tool**: Data Modeling Tool (e.g. Oracle SQL Developer Data Modeler)

Te_perform_code_improvement_by_guidance

- **Description**: Manual code improvement is essential to improve code quality, especially where tools cannot support it. Code Review can be a suitable starting point to begin manual improvement. In addition, there are guidelines for code refactorings that can support the user.
- **Guidance**: Code Improvement Guidelines (e.g. Guru Refactoring Guidelines)

Te_perform_code_improvement_by_tool

- **Description**: Semi-automated code improvement is technology dependent and there are available tools that support the user in this process. One well-known example is Code refactoring with tool support.
- **Tool**: Code Improvement Tools (e.g. Refactoring with ReSharper)

Te_perform_code_improvement_by_utility

- **Description**: Automated code improvement is technology dependent and in some technologies already possible to increase the quality of code without manual actions by the user.
- **Utility**: Code Improvement Utilities (e.g. Visual Assist)

Te_perform_data_model_quality_improvement_by_guidance

- **Description**: Guidelines and similar documents can support the user in improving data model quality. Depending on the technology there are guidelines or best practices available.
- **Guidance**: Data Model Quality Improvement Guidelines (e.g. XML schema design quality document)

Te_perform_data_model_quality_improvement_by_tool

- **Description**: Tools can support the quality improvement of data models. Depending on the technology there are tools available that improve the quality of data models including contradiction checks.
- **Tool**: Data Model Quality Improvement Tool (e.g. ERBuilder Data Modeler)

Te_perform_data_quality_improvement_by_guidance

- **Description**: There are guidelines available that support users by instructing quality improvements. These instructions can refer to manual jobs as well as to tools and utilities. Depending on the technology and kind of guideline, these guidelines can be on different abstraction levels.
- **Guidance**: Quality Improvement Guidelines (The Practitioner's Guide to Data Quality Improvement)

Te_perform_data_quality_improvement_by_tool

- **Description**: Various tools improve data quality semi-automatically by e.g. predefined patterns.
- **Tool**: Data Quality Improvement Tool (e.g. IBM InfoSphere QualityStage)

Te_perform_data_transformation_quality_analysis_by_tool

- **Description**: Data transformations are a kind of software. Hence, the quality of the software quality analysis tools can support users to analyze the quality of data transformations. Code Quality Tools also help to analyze the quality of data transformations.
- **Tool**: Software Quality Analysis Tool (e.g. SeaLights); Code Quality Tools (e.g. SonarQube)

Te_perform_detective_data_quality_analysis

- **Description**: Detective quality analytics improve the predictive and prescriptive analytics. There are a lot of detective data quality analysis tools available depending on the technology.
- **Tool**: Detective Data Quality Analysis Tool (e.g. Informatica Data Quality)

Te_perform_detective_data_quality_analysis_by_guidance

- **Description**: There are guidelines and best practices available that guide you through typical quality problems e.g. by questions or a list of problem types.
- **Guidance**: Quality Problems Guidance (e.g. Catalog of Quality Problems for Data, Data Models and Data Transformation)

Te_perform_explorative_data_quality_analysis

- **Description**: Explorative data analysis examines and appraises data of which there is little knowledge about their relationships. There are already approaches for tools to support users in explorative analysis.

- **Tool**: Explorative Data Analysis Tool (e.g. Pattern Tool)

Te_perform_sample_validation_by_utility

- **Description**: Validate an existing sample data set with a technology-dependent utility.
- **Utility**: Schema Validation utility (e.g. Saxon)

Te_perform_software_language_conformance_checks

- **Description**: Editors, software tools and compilers often have built-in language conformance checks after specifying the data format.
- **Utility**: Language Conformance Check Utility (e.g. Saxonica for XML)

Te_perform_software_vulnerability_checks

- **Description**: Code scanning supports users to analyze software code to find security vulnerabilities and related problems. This can be done automatically, regularly or by demand due depending on the technology.
- **Utility**: Software Vulnerability Check (e.g. GitHub Code Scan)

Te_release_data_model_by_guidance

- **Description**: Use of guidance to publish data models. E.g. the LIDO Profile workflow includes a workflow that is organized by the LIDO WG to publish existing LIDO Profiles (subschemas) of LIDO.
- **Guidance**: Defined Workflow for Publication of Data Models (e.g. LIDO Profile Workflow)

Te_release_data_transformation_by_guidance

- **Description**: Use of guidance to publish data transformation. Institutions often have portals or websites to publish existing data transformations. To our knowledge, publishing increases the likelihood of higher quality. In cases of popular data mappings, the transformation should be published on the data model web page to encourage reuse.
- **Guidance**: Defined Workflow for Publication (e.g. Research Data Publication Workflow by Bloom et al)

Te_requirements_review_by_guidance

- **Description**: A requirements review is a structured process that involves stakeholders and user groups that analyse the requirements document. They look for problems to ensure that the requirements are correct, complete, clear, and represent an understanding among all stakeholders.
- **Guidance**: Requirements Review Guideline (e.g. QRA: A Comprehensive Guide & Checklist)

Te_requirements_review_by_tool

- **Description**: There are quality tools that analyze the quality of the requirements specification: E.g. “The tools parse the requirements document based on the pre-defined glossary and provide a list of all occurrences of the weak-phrases in the document” [Denger and Olsson 2005]
- **Tool**: Requirements Review Tool (e.g. QVscribe for Quality Score for Requirements; QuARS (Quality Analyzer for Requirements Specifications); NASA Automated Requirement Measurement Tool)

Te_requirements_specification_by_tool

- **Description**: A Requirements Specification should contain the usual parts Current State, Target State, Stakeholders and requirements in a standardized form. The requirements should not contradict each other and should cover all relevant topics. Appropriate tools should guide the users to stick to form, content, structure and requirement language (e.g. IEEE[10]).
- **Tool**: Requirement Specification Tool (e.g. IBM Engineering Requirements Management DOORS Next - Best)

Te_use_code_project_qa

- **Description**: Code project tools usually include QA components that support users to improve code quality indirectly. This includes components like issue trackers, version control and release management (Conventional Commits, Semantic Versioning and Semantic Release).
- **Guidance**: Issue Tracker (e.g. GitHub Zenhub); Release Management (e.g. Conventional Commits)

Te_use_guidance_to_publish

- **Description**: Guidelines and policies for publishing increase availability, citability, reusability, and retrievability. This includes the standards for reuse and findability (e.g. DOI).
- **Tool**: Publish Guidelines (e.g. APA JARS)

Te_use_repositories_to_publish

- **Description**: Making your research available to a broader scientific community encourages collaboration and discussion across disciplines. There are institutional repositories, open-access repositories, and repositories operated by independent organizations available.
- **Tool**: Publication Repository (e.g. Zenodo)

[10] https://www.ieee.org/

Te_use_requirements_specification_technique

- **Description**: IEEE 830-1998 and IEEE 1233-1998 describe recommended practices and approaches for requirements specification. They also provide guidelines for compliance with these specifications.
- **Guidance**: Requirements Specification Technique (e.g. IEEE 830-1998 and IEEE 1233-1998)

Te_use_requirements_specification_templates

- **Description**: Requirement specification templates can help users efficiently create requirements in a standardized way.
- **Guidance**: Requirement Specification Templates (e.g. Klarity Business Requirement Template and IEEE Software Requirements Specification Template)

Te_use_publication_tools

- **Description**: Publication tools support the user in making the data available and retrievable. Data become publicly available, citable, identifiable, and retrievable, e.g. by adding metadata. Furthermore, the use of publication tools can improve transparency, verification and reproducibility. These aspects are essential in research and are very likely to increase citation rates.
- **Tool**: (Research) Publication Tools (e.g. DARIAH Publikator)

Te_validate_data_by_guidance

- **Description**: Organizations that employ people to collect data should have policies in place to support the collection. These guidelines are often in the form of documents or best practices and should be manually reconciled.
- **Guidance**: Editorial Guidelines (e.g. ULAN Editorial Rules)

Te_validate_data_by_tool

- **Description**: Organizations that employ people to collect data should have recommendations for tools in place to support the collection. These tools can be implemented to directly collect data or editors that are used to manually create data. These tools that focus on data collection or editors should have built-in validation.
- **Tool**: Data Editors (e.g. Oxygen); Data collection tool (e.g. Adlib)

Te_validate_data_by_utility

- **Description**: Some utilities support validation. For example, users can apply Schematron rules by using Saxon compilers. This makes sense, especially if the data model contains Schematron rules and the editor does not support Schematron.
- **Utility**: Validation compilers (e.g. Saxon for Schematron)

Te_verify_data_conformance_by_validation_guidance

- **Description**: There are conventions, standards that are available as guidelines. In particular, the rules that are not verifiable with utilities.

- Guidance: Data Model Documentation (e.g. LIDO Data Model Documentation)

Te_verify_data_conformance_by_validation_tool

- **Description**: Editors and other software can validate data for standard compliance, conventions and guidelines. For example, there are institution-specific policies that have been implemented as Schematron rules and can then be checked with an XML editor.
- **Tool**: Editor or Software with Data Model Validation (e.g. Oxygen for XML and XSD and Schematron validation)

Te_verify_data_conformance_by_validation_utility

- **Description**: Validation compiler can validate data for standard compliance, conventions and guidelines. For example, there are compilers that execute scripts e.g. for naming conventions like camelCase.
- **Utility**: Validation Compiler (e.g. Saxon-JS for Schematron)

Te_verify_data_language_conformance_by_guidance

- **Description**: Language validation is the process of checking data to confirm that it is both well-formed and valid in that it follows the structure defined by the language. Well-formed data follows the syntactic rules of the language. There
- **Guidance**: Language Guidelines (e.g. XML Constraints and Validation Rules)

Te_verify_data_language_conformance_by_tool

- **Description**: Data language validators check data for language conformance in terms of syntax, well-formedness, and other language rules. There are tools that have built-in validators available depending on the data language.
- **Tool**: Editor or Software with Data Language Validator (e.g. Atom in combination with linter-xmllint)

Te_verify_data_language_conformance_by_utility

- **Description**: Data language validators check data for language conformance in terms of syntax, well-formedness, and other language rules. There are automated validators available depending on the data language.
- **Utility**: Data Language Validator (e.g. XMLlint for XML)

Te_verify_data_model_design_language_by_guidance

- **Description**: Using standard model design languages improve the quality of data model design and the following data models. Language specifications like UML itself are guidelines to stick with these languages. There are also guidelines that are more user-oriented and guide users to comply with the language.
- **Guidance**: Data model design specification (e.g. UML); Data model design guidelines

Te_verify_data_model_design_quality_by_tool

- **Description**: Checking the quality of the data model semi-automatically using tools is already possible in some cases. Hence, depending on the modeling language, there are model verification tools that check for the presence of errors in the model. UCLAONT is an example of UML class diagram verification. Furthermore, structural analysis tools can use object-oriented measures such as design size, coupling, and complexity.
- **Tool**: Data Model Language Verification Tools (e.g. UCLAONT); Data Model Design Structure Analysis (e.g. SDMetrics)

Te_verify_data_model_design_language_by_utility

- **Description**: Checking data model designs and diagrams is crucial for ensuring quality like design specific constraints. There are tools available that check for compliance with e.g. constraints.
- **Utility**: Language Validation (IBM Rational Software Architect for UML)

Te_verify_data_requirements_by_guidance

- **Description**: Data that is to be collected needs to fulfill quality and general requirements. These requirements are defined by Requirements Specification which is often influenced by institutional guidelines. All data needs to be valid according to the data model and should stick to quality levels like traceability.
- **Guidance**: Data Model Validation; Quality Checklist (e.g. PUQI); Requirements Checklist; Guidelines for collection

Te_verify_data_transformation_design_quality_by_tool

- **Description**: The quality of data transformations is related to the quality of the data transformation design. Therefore, it is already important to analyze the design itself. Tools that support users to map models often include automated verification for completeness, contradictions, etc.
- **Tool**: Data Transformation Design Review (Apache Airflow)

Te_verify_data_model_design_language_by_utility

- **Description**: Verifying the correct use and conformance to data modeling languages like UML is essential for the quality of the data model design and further processing of it. Depending on the design language, the degree to which users can verify conformance differs.
- **Utility**: Data Model Design Language Verification Utility (e.g. UMLtoCSP)

Te_verify_rights_by_guidance

- **Description**: There are best practices and guidelines available that indicate standard or community recommended licenses and right statements. Some facilities and companies have licensing guidelines that employees must follow.
- **Guidance**: Rights and License Guidelines (e.g. OpenSource.org Software Licenses & Standards)

Te_verify_software_quality_by_guidance

- **Description**: There are best practices and guidelines available that support users to create software of high quality.
- **Guidance**: Software Quality Guidelines (e.g. ISO 5055 Software Quality Standards, The Twelve-Factor App)

Te_verify_software_quality_by_tool

- **Description**: Besides software testing tools, there are also tools available that test the quality of a software software.
- **Tool**: Software Quality Tool (e.g. CESSDA Software Maturity Levels Test)

Skills

The process describes skills as part of tasks. The skills are learned abilities to perform a task and are identified by the prefix “Sk_” and the name of the skill. All skills are defined using a uniform structure, which is defined as follows:

Sk_[*name of skill*]

- **Description**: A mainly informative form of presentation that describes the skill in a fact-based manner.

Each skill is defined and described separately and is linked via the tasks and the roles. The following list of skills is sorted by alphabetical order.

Sk_analzye_compliance

- **Description**: The ability to identify standards and recommendations and check whether certain content meets these (compliance).

Sk_analytic_thinking

- **Description**: The ability to identify problems, organize them and develop solutions.

Sk_analyzing_data_integrity

- **Description**: The ability to understand sources, metadata and guidelines of data as well as to review assurance, accuracy and consistency of data.

Sk_analyze_quality

- **Description**: The ability to analyze the quality of inputs. Two types of quality analysis are common: assessment and evaluation. The ability includes the knowledge of methods and techniques required for the specific use case.

Sk_data_design

- **Description**: The ability to define data design (type, format, volume), data implementation and being aware of any system that processes data.

Sk_data_management

- **Description**: The ability to analyze and define dependencies of data and responsibilities.

Sk_data_model_design

- **Description**: The ability to understand and analyze data models and define data model designs

Sk_data_modelling

- **Description**: The ability to understand the structure of data models and to develop a mapping of information.

Sk_data_transformation_design

- **Description**: The ability to understand and analyze software designs, specifically for data transformations, and define data transformation designs

Sk_define_requirements

- **Description**: The ability to define requirements to create a requirements specification. This includes dividing functional and nonfunctional requirements and the ability to define a target state and a current state.

Sk_documentation

- **Description**: The ability to empathize with users and create user-focused documentation.

Sk_execute_script

- **Description**: The ability to execute a developed script in the required environment. This often requires the operation of a command line tool.

Sk_generic_data_model_knowledge

- **Description**: The knowledge about a data model and the related data. This includes the context, possibilities and limitations of this data model.

Sk_general_domain_knowledge

- **Description**: The knowledge of problems and solutions of “a specific, specialized discipline or field”. (Wikipedia)

Sk_programming

- **Description**: Ability to programming, also known as coding skills, to create code in a specific programming language.

Sk_specific_data_model_knowledge

- **Description**: The knowledge that extends generic data model knowledge by knowledge about all data elements.

Sk_specific_domain_knowledge

- **Description**: The knowledge that extends the general domain knowledge by understanding a specific topic as part of a discipline or field. This includes identifying problems and developing solutions for these. It often requires experience for a couple of years to classify the importance and accuracy of information and problems.

Sk_structural_thinking

- **Description**: The ability to understand and develop complex structures.

Sk_use_of_data_mapping_tool

- **Description**: The ability to operate mapping tools that create data transformations.

Roles

The process describes roles separately and they are indirectly linked to the tasks as they contain skills. The roles are identified by the name of the role. All roles are defined using a uniform structure, which is defined as follows:

[*name of role*]

- **Description**: A mainly informative form of presentation that describes the skill in a fact-based manner.
- **Skill**: An ability that should be learned by the person who performs this role.

There are several people involved in a quality management process. These people have different backgrounds and perform different tasks and responsibilities. In order to better understand the people involved and their tasks, defining roles and responsibilities is important. KONDA defined roles and responsibilities that are usually involved in a data quality management process for the scope that is defined above. These definitions are based on literature research and the expert workshops and interviews described in *Section Approach*. The number of roles in such a process depends on the organization size and, consequently, on the amount of data it manages. Generally, people can take multiple roles, one role can be taken by multiple persons and some roles might not be filled at all. Each role is defined and described separately and is not directly linked. The following list of roles is sorted by alphabetical order.

Data analyst

- **Description**: Data analysts (also called data strategists) explore, assess and summarize data and reports by performing statistical analyses of data. They develop and execute trend-pattern analytics plans to create new knowledge and to help with (business) decisions. Tasks are the implementation of (new) systems or software for data collection, obtaining correct data from external and internal databases and defining correct key figures, on which e.g. the profitability of individual company processes can be read.
- **Skills**: Sk_analytic_thinking; Sk_define_requirements; Sk_general_domain_knowledge; Sk_structural_thinking; Sk_analyze_quality; Sk_data_transformation_design; Sk_execute_script; Sk_programming

Data collector

- **Description**: Data collectors (also called data producers and data originators) are individuals or organizations which capture/produce/generate data in all kinds of ways like using software or hardware with different purposes (with or without consent). This includes digitization, documentation and data management (e.g. use of cameras, creating metadata and updating or deleting data). As such, data collectors are responsible for the quality of data within the source system or scope to which they have access. They ensure that data complies with data consumers' quality requirements, but they are not necessarily accountable for it.
- **Skills**: Sk_define_requirements; Sk_execute_script; Sk_general_domain_knowledge; Sk_specific_domain_knowledge; Sk_documentation

Data consumer

- **Description**: Data consumers (also called data users) are people like researchers, organizations and platforms that buy or get free access to data, usually through applications or websites. A regular user defines data quality standards and reports errors to the responsible persons. The consumers take the end product created from the data and use it for purposes such as research, planning, forecasting and decision-making. The end product could be reports, information processed by functions or research results. Consumers know what to expect in terms of quality and it is important that the data meets their requirements. But they have to be responsible for defining these requirements.
- Skills: Sk_define_requirements;

Data manager

- **Description**: Data custodians (also called data curators and data custodians) manage the technical environment of data maintenance and storage. They lead the technical design, implementation and continued maintenance of data, data models and data transformation. This includes ingest, archive retrieval, data access and file system and technology upgrade. Their responsibilities also include improving data accessibility, discoverability and interoperability. Generally, a data custodian ensures the quality, integrity and safety of data in a data lifecycle.
- **Skills**: Sk_analytic_thinking; Sk_structural_thinking; Sk_data_design; Sk_data_management; Sk_data_modelling; Sk_define_requirements;

Sk_execute_script; Sk_generic_data_model_knowledge;
Sk_use_of_data_mapping_tool; Sk_specific_data_model_knowledge;
Sk_programming; Sk_general_domain_knowledge; Sk_documentation;
Sk_data_transformation_design; Sk_data_model_design; Sk_analyze_quality;
Sk_analyzing_data_integrity

Data owner

- **Description**: Data owners control and manage the quality of one or more given datasets, specifying data quality, data model and data transformation requirements. They are generally responsible for the quality and security of the data lifecycle within their specific subject area (e.g. Risk Finance, or Customer). They also define data requirements and policies for data collection and use based on reports of data stewards and data analysts.
- **Skills**: Sk_analytic_thinking; Sk_analyzing_data_integrity; Sk_data_design;
  Sk_data_management; Sk_data_modelling; Sk_define_requirements;
  Sk_generic_data_model_knowledge; Sk_general_domain_knowledge;
  Sk_structural_thinking; Sk_analzye_compliance; Sk_analyze_quality;
  Sk_data_design; Sk_data_model_design; Sk_data_transformation_design;

Data steward

- **Description**: Data stewards (also called compliance specialists and scientific specialists) are in charge of content, context and associated rules and standards of data, data models and data transformations. They are usually skilled experts in terms of data and document guidelines for data and metadata generation, access and use. This makes them responsible for the quality of a defined dataset on a day-to-day basis. Data stewards can also advise on how to improve existing data practices. They use regular monitoring of data quality and provide reports.
- **Skills**: Sk_analytic_thinking; Sk_analyzing_data_integrity; Sk_data_design;
  Sk_data_modelling; Sk_define_requirements; Sk_generic_data_model_knowledge;
  Sk_general_domain_knowledge; Sk_specific_domain_knowledge;
  Sk_structural_thinking; Sk_use_of_data_mapping_tool;
  Sk_general_domain_knowledge; Sk_specific_data_model_knowledge;
  Sk_structural_thinking; Sk_use_of_data_mapping_tool; Sk_documentation;
  Sk_analzye_compliance

Artifacts

The process describes artifacts in the form of inputs or outputs of an activity. An artifact as an input is a necessary work product that is required for a particular activity. An artifact as an output is a required work product that is demanded as the completion of a particular activity. The artifacts are identified by the prefix “Ar_” and the name of the artifact. All artifacts are defined using a uniform structure, which is defined as follows:

Ar_[*name of artifact*]

- **Description**: A mainly informative form of presentation that describes the skill in a fact-based manner.

Each artifact is defined and described separately and is linked via the skills to the tasks. The following list of artifacts is sorted by alphabetical order.

Ar_PublicAccess

- **Description**: A public access to produced artifacts. This includes the use of persistent identifiers, metadata and links to associated works.

Ar_PublishedDataset

- **Description**: A published data set that inherits from Ar_PublicAccess and focuses on data sets.

Ar_PublishedDataModel

- **Description**: A published data model that inherits from Ar_PublicAccess and focuses on data models.

Ar_PublishedDataTransformation

- **Description**: A published data transformation that inherits from Ar_PublicAccess and focuses on data transformations.

Ar_DataDesign

- **Description**: A definition of the data structure and methods to generate and process data.

Ar_DataModel

- **Description**: In the context of semi-structured data, a data model is also called schema. This artifact is not published, unlike Ar_PublishedDataModel.

Ar_DataModelDesign

- **Description**: A data model design defines the allowed expressions in an artificial 'language' with a scope that is limited by the scope of the model. (Wikipedia)

Ar_DataModelMapping

- **Description**: A Mapping of the data elements between two data models. Here, the result is the assignment of source and target fields. This should ideally include all elements and their properties, and may also include constraints such as if-then-else relationships. It should also contain all the information about the two data models necessary for the transformation.

Ar_DataSet

- **Description**: A data set “is a collection of data.” (Wikipedia). This artifact is not published, unlike Ar_PublishedDataset.

Ar_DataTransformation

- **Description**: In computing, data transformation is the process of converting data from one format or structure into another format or structure. (Wikipedia). This artifact is not published, unlike Ar_PublishedDataTransformation.

Ar_QualityReport

- **Description**: A quality report is a result of a quality analysis that contains the exploration and verification of data, data model, or data transformation. Depending on the quality analyses, the report may contain different aspects. In any case, it should provide users with indications of poor quality, but also of good quality, as a kind of high-quality certificate. The type of detail level can also vary. Also, indications of poor quality of any value might be helpful.

Ar_DataQualityReport

- **Description**: A data quality report that inherits from Ar_QualityReport.

Ar_DataModelQualityReport

- **Description**: A data model quality report that inherits from Ar_QualityReport.

Ar_DataTransformationQualityReport

- **Description**: A data transformation quality report that inherits from Ar_QualityReport.

Ar_RequirementSpecification

- **Description**: “A Requirement Specification is a collection of the set of all requirements that are to be imposed on the design and verification of the product. The specification also contains other related information necessary for the design, verification, and maintenance of the product.” (reqexperts)

Ar_RequirementSpecificationOfData

- **Description**: A requirements specification of the data. This inherits from Ar_RequirementSpecification.

Ar_RequirementSpecificationOfDataModel

- **Description**: A requirements specification of the data model. This inherits from Ar_RequirementSpecification.

Ar_RequirementSpecificationOfDataTransformation

- **Description**: A requirements specification of the data transformation. This inherits from Ar_RequirementSpecification.

Ar_UnmetRequirements

- **Description**: A subset list of the requirements specification with requirements that are not yet fulfilled.

Ar_UnmetRequirementsOfData

- **Description**: A list of unmet data requirements based on the requirements specification of data. This inherits from Ar_UnmetRequirements.

Ar_UnmetRequirementsOfDataModel

- **Description**: A list of unmet data model requirements based on the requirements specification of data model. This inherits from Ar_UnmetRequirements.

Ar_UnmetRequirementsOfDataTransformation

- **Description**: A list of unmet data transformation requirements based on the requirements specification of data transformation. This inherits from Ar_UnmetRequirements.

Instantiation Context

The process is a result of a three-year research project, called KONDA, which focussed on the quality of research data. KONDA is an interdisciplinary joint research project at the Universities of Göttigen and Marburg, funded by the Federal Ministry of Education and Research (BMBF) for three years, from 2019 to 2022. The acronym “KONDA” stands for continuous quality management of dynamic research data on objects of material culture using the LIDO standard.
The project intends to develop a systematic quality assurance of structured research data on objects of material culture, a desideratum for research in the humanities and cultural sciences. The approach of a continuous quality management process (QM process) differentiating according to data, data models and data transformations over the entire lifecycle of data is groundbreaking and pioneering. Therefore, a generic QM process for dynamic, partly uncertain research data will be developed. This process is applied to the internationally accepted harvesting format Lightweight Information Describing Objects (LIDO) for objects of material culture. It is then converted into specified curatorial criteria for data generation and curation of art historical research data describing various genres of material objects that are collected e.g. in museums or university collections.
The QM Process is to be tested on selected databases e.g. of the Deutsches Dokumentationszentrum für Kunstgeschichte – Bildarchiv Foto Marburg and the University Collections of the University of Göttingen. The resulting QM processes are documented in manuals and will be made available to the professional community.

Affiliates:

- Philipps-Universität Marburg / Deutsches Dokumentationszentrum für Kunstgeschichte – Bildarchiv Foto Marburg (DDK)[11]
- Göttingen State and University Library[12]
- Philipps-Universität Marburg / Department of Mathematics and Computer Science[13]

[11] https://www.uni-marburg.de/de/fotomarburg
[12] https://www.sub.uni-goettingen.de/
[13] https://www.uni-marburg.de/fb12/en/index_html?set_language=en

Bibliography

[Abiteboul 1997] S. Abiteboul, „Querying semi-structured data“, in International Conference on Database Theory, Springer, 1997, S. 1–18.

[Ambika 2020] P. Ambika. 2020. Machine learning and deep learning algorithms on the Industrial Internet of Things (IIoT). In Advances in Computers. Vol. 117. Elsevier, 321–338. https://doi.org/10.1016/bs.adcom.2019.10.007

[Aurum and Wohlin 2005] Aurum, Aybüke, und Claes Wohlin. Engineering and Managing Software Requirements. Berlin: Springer, 2005.

[Bogner et al. 2014] A. Bogner, B. Littig, and W. Menz, Interviews mit Experten. Eine praxisorientierte Einführung. 2014.

[Brinkkemper 1996] S. Brinkkemper, “Method engineering: engineering of information systems development methods and tools,” Information and Software Technology, vol. 38, no. 4, pp. 275–280, Jan. 1996, doi: 10.1016/0950-5849(95)01059-9.

[Denger and Olsson 2005] C. Denger and T. Olsson, “Quality Assurance in Requirements Engineering,” in Engineering and Managing Software Requirements, A. Aurum and C. Wohlin, Eds., Berlin/Heidelberg: Springer-Verlag, 2005, pp. 163–185. doi: 10.1007/3-540-28244-0_8.

[Deutsche Forschungsgemeinschaft 2015] Deutsche Forschungsgemeinschaft, “Leitlinien zum Umgang mit Forschungsdaten.” 2015.

[Engels and Sauer 2010] G. Engels and S. Sauer, “A Meta-Method for Defining Software Engineering Methods,” in Graph Transformations and Model-Driven Engineering, G. Engels, C. Lewerentz, W. Schäfer, A. Schürr, and B. Westfechtel, Eds., in Lecture Notes in Computer Science, vol. 5765. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 411–440. doi: 10.1007/978-3-642-17322-6_18.

[Fouché et al. 2010] C. Fouché and G. Light, “An Invitation to Dialogue: ‘The World Café’ In Social Work Research,” Qualitative Social Work, vol. 10, no. 1, pp. 28–48, Mar. 2011, doi: 10.1177/1473325010376016.

[Gutzwiller et al. 1994] Gutzwiller, Thomas A. Das CC RIM-Referenzmodell für den Entwurf von betrieblichen, transaktionsorientierten Informationssystemen. Bd. 54. Betriebs- und Wirtschaftsinformatik. Heidelberg: Physica-Verlag HD, 1994. https://doi.org/10.1007/978-3-642-52405-9.

[Kesper at al. 2023] Arno Kesper, Markus Matoni, Julia Rössel, Gabriele Taentzer, Michelle Weidling, & Viola Wenz. (2023). Catalog of Quality Problems in Data, Data Models and Data Transformations (Version 2). Zenodo. https://doi.org/10.5281/zenodo.7757293

[Moody and Shanks 1994] Daniel L. Moody and Graeme G. Shanks. 1994. What Makes a Good Data Model? Evaluating the Quality of Entity Relationship Models. In

Entity-Relationship Approach - ER’94, Business Modelling and Re-Engineering, 13th International Conference on the Entity-Relationship Approach, Manchester, UK, December 13-16, 1994, Proceedings (Lecture Notes in ComputerScience,Vol.881), PericlesLoucopoulos(Ed.). Springer, 94–111. https://doi.org/10.1007/3-540-58786-1_75

[Ray 2014] J. M. Ray, Research Data Management: Practical Strategies for Information Professionals. Purdue University Press, 2014.

[RFII 2019] German Council for Scientific Information Infrastructures (RfII), “The Data Quality Challenge. Recommendations for Sustainable Research in the Digital Turn,” Göttingen, 2020. [Online]. Available: https://rfii.de/?p=4203

[Schwaber and Beedle 2002] K. Schwaber and M. Beedle, Agile software development with Scrum, Pearson international ed. in Series in agile software development. Upper Saddle River, N.J.: Pearson Education International, 2002.

[Shuja and Krebs 2008] A. K. Shuja and J. Krebs, IBM Rational Unified Process reference and certification guide: solution designer. Upper Saddle River, NJ: IBM Press/Pearson, 2008.

[West 2010] M. West, Developing High Quality Data Models. 2010.