FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Ein visueller Ansatz zur Exploration von Datenqualitätsproblemen in multivariaten und zeitorientierten Daten

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Wirtschaftsinformatik

eingereicht von

Thomas Ziegelbecker, Bsc
Matrikelnummer 0925321

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof. Dr. Mag. Silvia Miksch
Mitwirkung: Dr. Dipl.-Ing. Bilal Alsallakh
Dipl.-Ing. Markus Bögl
Dipl.-Ing. Christian Bors

Wien, 25. August 2016

Thomas Ziegelbecker   Silvia Miksch

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

A Visual Approach for Exploring Quality Problems of Multivariate and Time-Oriented Data

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Business Informatics

by

Thomas Ziegelbecker, Bsc
Registration Number 0925321

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof. Dr. Mag. Silvia Miksch
Assistance: Dr. Dipl.-Ing. Bilal Alsallakh
Dipl.-Ing. Markus Bögl
Dipl.-Ing. Christian Bors

Vienna, 25th August, 2016

Thomas Ziegelbecker   Silvia Miksch

Erklärung zur Verfassung der Arbeit

Thomas Ziegelbecker, Bsc
Krongasse 22/5, 1050 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 25. August 2016

Thomas Ziegelbecker

Acknowledgements

First, I wish to thank my supervisors for their patience, their understanding and all their valuable inputs throughout the numerous session. I am also grateful to Sara as well as to all my best friends for their encouragement and guidance throughout the process of writing this. I particularly have to thank all the inspiring lecturers at Vienna University of Technology as well as abroad at Kungliga Tekniska Högskolan who shaped me as an individual and provided me with experiences that I will always remember for the rest of my life.

I allow myself to finish this with one of my most favorite quotes.

> It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to Heaven, we were all going direct the other way – in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only. [Dic12]

Kurzfassung

Bei annähernd jedem Schritt den wir heutzutage tätigen, werden Daten gespeichert. In vielen Fällen handelt es sich dabei allerdings um fehlerhafte Daten, wobei die Gründe dafür sehr vielfältig sein können. Diese reichen einerseits von Mängeln, wie beispielsweise Messungenauigkeit, über Messfehler bis hin zu menschlichem Versagen. Problematisch werden die fehlerhaften Daten vor allem dann, wenn zum Beispiel Analysten unwissentlich ihre Entscheidungen auf deren Basis treffen. Mögliche Konsequenzen aus diesen Entscheidungen können beispielsweise falsche Schlussfolgerungen sein, die wiederum zu höheren Kosten führen können. Eine Möglichkeit dieses Problem zu adressieren, ist die Qualität der fehlerhaften Daten zu visualisieren, um damit bei den Entscheidungsträgern ein Bewusstsein für die Problematik zu schaffen. Darüber hinaus haben Forschungsergebnisse gezeigt, dass die Visualisierung von Datenqualitätsproblemen die Entscheidungsfindung verbessert. Trotz dieser Beobachtung wurde bisher auf dem Gebiet der Datenqualitätsvisualisierung von uni- und multivariaten Daten nur wenig Forschung betrieben.

Der Schwerpunkt dieser Arbeit liegt aus den genannten Gründen auf der Visualisierung von Datenqualitätsproblemen und ihrer Integration in den Datenexplorationsprozess. Das primäre Ziel dabei ist, einen neuen Ansatz zu finden, der die Qualitätsprobleme von multivariaten und zeitorientierten Daten sowohl im Überblick als auch im Detail darstellt. Zu diesem Zweck wird ein Domänenproblem aus der Bohrindustrie herangezogen. Bei den von mehreren Sensoren zur Verfügung gestellten Daten handelt es sich um Bohrdaten, welche unter anderem fehlende Werte, ungültige Werte und Ausreißer enthalten. Zur Lösung dieser Probleme werden die vorhandenen Visualisierungsmöglichkeiten bewertet und auf deren Grundlage Gestaltungsmöglichkeiten entwickelt um Datenqualitätsprobleme sowohl im Überblick als auch im Detail darzustellen. In einem nachfolgenden Schritt werden die zuvor getroffenen Designentscheidungen im Rahmen eines Prototyps implementiert und im Zuge von Experteninterviews evaluiert.

Die Ergebnisse dieser Interviews werden zusammengefasst, diskutiert und als Argumentationsgrundlage für zukünftige Designentscheidungen festgehalten. Darüber hinaus bieten die Ergebnisse auch Argumente für bestimmte Interaktionstechniken sowie Einblicke in die zur Implementierung eingesetzten Algorithmen und Technologien. Die abschließenden Ergebnisse lassen Schlussfolgerungen hinsichtlich der Auswahl von Ansätzen zur Visualisierung von Datenqualitätsproblemen zu und bilden die Grundlage für weitere Forschung.

Abstract

Today we produce and capture data at almost each and every step. In many cases, this data is imperfect, due to various defects such as sensor variability, errors in measurement, or by human error. Analysts and decision makers unknowingly base their decisions on such imperfect data, which often leads to poor decisions and high costs. One way to address this problem is to visualize data quality problems to make decision makers more aware of them. Despite existing literature proving that data quality visualization improves decision-making, only little research has been conducted in the field of univariate and multivariate data quality visualization.

Therefore, the focus of this work will be on incorporating data quality visualization into the data exploration process, where the main contribution is to provide a novel approach for visualizing data quality problems of multivariate time-oriented data in both, overview and detail. For this purpose, a particular domain problem from the drilling industry will be used. The data itself is provided from multiple sensors that transmit time-stamped raw drilling-data, which contains data quality problems such as missing values, invalid values and outliers.

In this work I examine existing data quality visualizations for multivariate time-oriented data. Based on this literature research I develop and discuss several design options in overview and detail for visualizing the data quality problems identified in combination with the domain problem. In a subsequent step I implement selected design approaches in a prototype and evaluate them in the context of expert interview sessions. The results of these session are then reported and discussed, providing further rationales for the design choices made. In addition, the results also provide arguments for specific interaction techniques (i.e., combined interactive views) as well as they offer insights into algorithms and technologies used. Overall, the results give conclusions for selecting data quality visualization approaches and make suggestions for further research areas such as the aggregation algorithms for data quality problems.

Contents

CHAPTER 1

Introduction

1.1 Problem Statement

We live in a more and more data driven world, where people produce and capture data at almost each and every step. For instance, only mobile data traffic has grown 4,000-fold over the past 10 years and almost 400-million-fold over the past 15 years [For16]. In many cases, this data is imperfect, due to various defects such as sensor variability, errors in measurement, or by human error. Analysts and decision makers unknowingly base their decisions on such imperfect data, which often leads to poor decisions and high costs [KPP$^+$12]. This fact is further underlined by resources claiming that "*data quality is an important aspect to the value of a visualization*" [Sti13, Joh04, Sha05]. One way to overcome this problem is to take measures to improve the recording quality of the data (e.g., by regularly checking the calibration of the sensor) [Red98, Eck02]. Another way is to convey data quality to make decision makers more aware of data quality problems. The focus of this work will be on the latter, where data quality problems are excepted and incorporated into the data exploration process [XHWR06].

Although methods have been developed to identify and indicate data defects, errors are often context dependent and require further human judgment [KPP$^+$12]. As a result, recent work proposed to combine automated methods and appropriate data visualization together with human judgment [SEG05]. Despite existing literature arguing that data quality visualization improves decision-making, only little research has been conducted in the field of univariate [SEG05, KPP$^+$12] and multivariate [XHWR06, ZWRS07] data quality visualization.

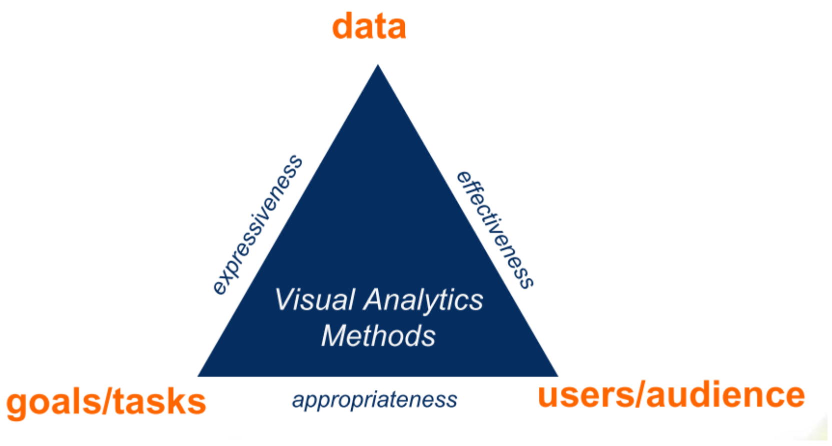


Figure 1.1: User-Centered Design Triangle [MA14].

In the following subsections I conduct a problem analysis, which is based of the design triangle of Miksch and Aigner [MA14]. This approach follows the principles of user-centered design (see Figure 1.1), while asking the following three key questions:

1. Who are the users of the system? **(User)**

2. What kind of data are the users working on? **(Data)**

3. What are the general tasks of the user? **(Tasks)**

1.1.1 The User

The users are data quality engineers in the drilling industry, who are responsible for inspecting sensor measurements. They assess automatically segmented sensor data and, in some cases, have to reassign wrongly segmented labels. In other cases, a data quality engineer has to remove certain records completely, because he or she identified them as outliers. Furthermore, these users are experienced with analyzing segmented data and take qualitative decisions when faced with irregular data values. Hence, their primary goal is to efficiently find and select data with irregularities in order to draw conclusions based on this data.

1.1.2 The Data

In this work I deal with time-stamped raw sensor data, which is obtained from measurements of multiple sensors over time at a fixed interval T ($T = 15s$). Given the fact that this data is imperfect, the following data quality issues were identified:

- missing values
- missing timestamps (i.e, timestamp gaps > T)
- invalid values (specified as negative values)
- outliers (extreme values)

For this work three different data sets were provided by the supervisor (see Table 1.1)

Table 1.1: Provided Data Sets.

Name	**Size**	**Interval**
01_raw_data_set1.csv	44 MB	6 weeks
02_raw_data_set2.csv	164.8 MB	3 months
02_labels_data_set2.csv	28 MB	3 months

The only difference between the first and the second dataset is an additional sensor (channel) that has been added to the latter. The following Tables 1.2,1.3 and 1.4) describe the data sets in more detail.

Table 1.2: 01_raw_data_set1.csv - data description.

Name	**Type**	**Description**
Timestamp	long	a Java timestamp provided as a long value
h	Numeric	
f	Numeric	
m1	Numeric	
m2	Numeric	
p1	Numeric	
r	Numeric	
t	Numeric	
p2	Numeric	
w	Numeric	

1.1.3 The Task

The data quality engineers have the following tasks.

- *T1: As data quality engineer [User], I want to get the missing data gaps in time series [Data] visualized. So I can easily identify where the gaps are located on*

Table 1.3: 02_raw_data_set2.csv - data description.

Name	Type	Description
Timestamp	long	a Java timestamp provided as a long value
h	Numeric	
f	Numeric	
m1	Numeric	
m2	Numeric	
p1	Numeric	
r	Numeric	
t	Numeric	
p2	Numeric	
r2	Numeric	additional channel (compare to data set 1 1.2)
w	Numeric	

Table 1.4: 02_labels_data_set2.csv - data description.

Name	Type	Description
startTimestamp	long	a Java timestamp provided as a long value
endTimestamp	long	
startm2	Numeric	
endm2	Numeric	
startm1	Numeric	
endm2	Numeric	
value	Numeric	the actual label value, will be mapped

different zooming levels [Task]. Furthermore, it should be clear signs to the channels that have gaps or invalid data (please consider all the negative values as invalid data and null data as missing data). Also the difference between the missing timestamps and the missing values should be visually discovered

- *T2: As data quality engineer [User], I want to get all the outliers in sensors data [Data] visualized on different zooming levels and in which channels these outliers exist [Task] (the outlier is the extreme positive or negative value)*

From these two tasks I conclude that data quality engineers mainly deal with missing data, invalid data, missing timestamps and outliers. Moreover, all these data quality issues occur in different channels and at different zoom levels.

1.2 Motivation

The primary motivation behind this work is to solve a specific business problem in the drilling industry, which deals with multivariate and time-series data. The data, provided by the partner company, represents different activities, which are performed during a drilling process. One such data record is composed of many different measurements provided by multiple sensors at different times. The main concern of the partner company is to automatically segment the data by using a rule-based system that determines activities for a certain time interval. Finally, this segmented data reconstructs the drilling process and is then used to derive actions to improve the quality of the process. This problem was first introduced in Alsallakh et al. [ABG+14], where they propose a visual analytics approach that provides an analyst with four interactive views for inspecting the segmentation results and identifying mislabeled segments (see Figure 1.2). More particularly, they combined a slider to traverse time (a), a line chart to visualize the values provided by the sensors (b), a control panel to adopt for instance thresholds of the segmentation algorithms (c) and a bar to visualize the segmentation results (d).

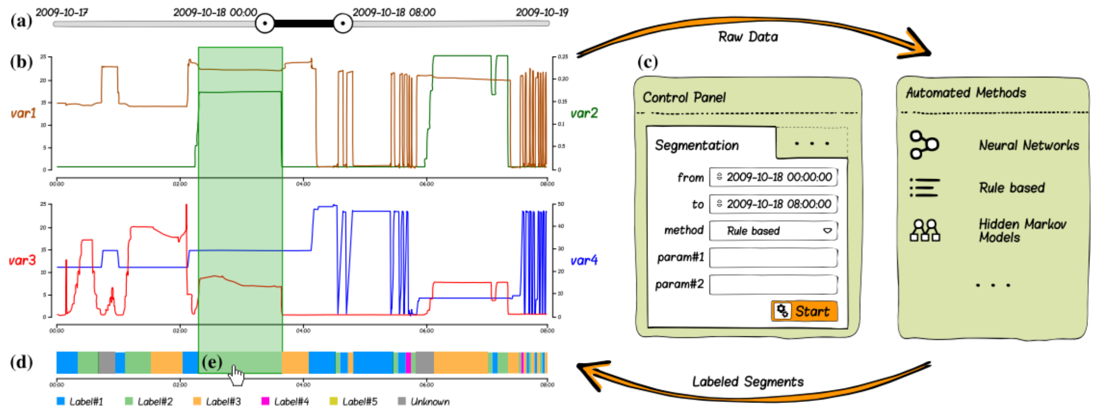


Figure 1.2: Visual Analytics approach - a tool to rate and display data quality [ABG+14].

Their main idea is to use visualization to inspect the results, adopt the algorithms, and to correct bad segmentation results computed by automated algorithms. After some iterations the correctly segmented data is the output that is then used for latter in depth analysis. However, what does all that have to do with data quality visualization? In short, Alsallakh et al. [ABG+14] named data quality visualization as one of the possible extensions to their approach, which is why this business case was picked, as it suits the identified research gap. Therefore, the primary motivation of this work is to use the described business case in order to find generic solutions to data quality visualization in the context of multivariate and time-series data.

1.3 Aim of Work

Research Questions

The main contribution of this work will be a novel approach for visualizing data quality problems of multivariate time-oriented data incorporated into the exploration process. Therefore, the following corresponding research questions were identified:

1. How to provide an overview visualization of data quality of large multivariate time-oriented data.

2. How to provide detailed visualization of data quality of multivariate time-oriented data for a specific time interval.

3. How effective are the proposed techniques when applied in practical applications.

For this purpose, a particular domain problem from the drilling industry[1] will be used. The data itself is provided from multiple sensors that transmit time-stamped raw drilling-data, where the following data quality problems occur:

- Missing values (in the .csv)
- Missing time stamps (i.e., timestamp gaps)
- Erroneous values (i.e., invalid values, specified as negative values)
- Extreme values (i.e., very high or low values - outliers)

Derived from the problem and the provided data the following artifacts are expected to be generated in the context of this work:

1. A novel approach to effectively visualize data quality of multivariate and time-oriented data, both in overview and in detail.

2. A prototype implementation of the proposed approach, where the main focus is placed on the visualization of data quality indicators, rather than the underlying algorithms that compute these indicators.

3. An evaluation that will be used to asses the effectiveness of the proposed approach.

[1]Further details about the problem domain can be found in Alsallakh et al. [ABG+14]

1.4 Methodological Approach

The work is based on the design-science paradigm (DSP) [vAMPR04], as well as it follows the principles of the nine-stage design study methodology (DSM) of Sedlmair et al. [SMM12]. The focus of design science is to build and evaluate new innovative IT solutions, which are designed to meet defined business needs, while the main goal is on providing utility. Design-science, in its essence, is a problem solving process that tries to *solve unsolved problems in unique or innovative ways, or solved problems in more effective or efficient ways.* DSM refines those ideas and offers a nine-stage framework tailored for design studies conducted in the field of information visualization and visual analytics. In the following subsections, all the steps necessary in order to contribute to the *knowledge base* are elaborated in more detail, where each step is denoted by its respective stage of the DSM.

1.4.1 Design (Precondition)

According to DSM, in the first stage (*learn*) knowledge of visualization techniques and methods is very important, as it informs all latter stages. Therefore, a state-of-the-art literature research will be the first step in order to gather the necessary knowledge (e.g., outlier visualization techniques) related to the domain problem. As this work will be a contribution related to Alsallakh et al. [ABG$^+$14], most work contained in the other two stages (*winnow* and *cast*) has already been done (e.g., identify collaborator roles).

1.4.2 Build (Core)

First, a problem characterization and abstraction of the domain problem - derived from Alsallakh et al. [ABG$^+$14] - will be provided (*discover*). In the following stage (*design*) generation and validation of data abstractions, visual encodings and interaction mechanisms will be defined. For instance, aggregation will be used as a main data reduction method to handle large time-oriented data. In the matter of visual encoding and interaction the design will be primarily based on the findings of Aigner et al. [AMST11, TA13], as well as on the resources found during the precondition phase. The resulting consideration space (techniques, methods and tools) will then be filtered down to a narrow proposal space, which is used for discussion with an expert (i.e supervisor). The feedback of the expert is then used to select a solution that will be implemented as a prototype. During the next stage (*implement*) rapid prototyping and agile methods, as well as usability inspections will be used, as recommended by the DSM. Finally, in the last stage (*deploy*), the prototype will be deployed and evaluated using informal evaluation. Additionally, the whole build process will follow the user-centred-design approach and the design triangle, as proposed by Miksch and Aigner [MA14].

1.4.3 Evaluate (Analyis)

The feedback obtained in the previous phase is then used to either confirm, or reject the effectiveness of the proposed solution. Another possible outcome of the analysis of the findings could be the refinement of existing guidelines, or the creation of completely new ones (*reflect*).

CHAPTER 2

State-of-the-Art

In the following chapter data quality visualization approaches will be examined. First, the research method of how all the different examples and solutions were acquired is introduced in Section 2.1. Then a concise introduction to *data quality* and its different dimensions is given in Section 2.2 before the discovered data quality visualizations are presented in Section 2.3. These visualizations are then categorized into three different groups, which can be found in Section 2.1.2. Additionally, specific data quality visualization approaches are presented in Section 2.4). As a result of this research, the conclusions drawn from a comparison of the different visualization approaches are presented in Section 2.5.

2.1 Method

This section describes how the references used in the state-of-the-art research were acquired and selected. This research was conducted in the time period from April 2014 till the end of June 2014. The following steps were taken throughout the research:

1. The problem (see Section 1.1) was defined on the basis of Alsallakh et al. [ABG^{+}14] and on additional presentation slides provided by the partner.
2. Keywords were derived for later use during search (see Table A.1)
3. Desired search engines were selected
4. A set of criteria was defined for the selection
5. The search was carried out
6. The results were categorized into different groups
7. The results were compared and conclusion was drawn that only few resources exist in combination with multivariate and time-oriented data.

2.1.1 Search Engines

The following list depicts all the search engines that were used during the actual search:

- VUT *Catalog Plus* (including licensed articles from ACM, IEEE or SpringerLink)
- *ACM digital library* and *IEEE Xplore* - for searching with temporally refined parameters
- Google Scholar
- Microsoft Academic Search
- Google Web Search

2.1.2 Search Process

The search process itself was carried out within the following 4 iterative rounds:

1. Overview: Search with very narrow search terms like *data quality visualization with multivariate time oriented data.* As this search did not come up with useful results the primary search term was refined for the first time to *data quality visualization.*
2. Keyword Refinemfent: all engines were queried with the refined short form of *data quality visualization*
3. Temporal Refinement: the search results were restricted by the time period they were from (i.e., from 2010 to 2014), as only little was found in the first round.
4. Details: additionally all search engines were queried for keywords that were derived from the different quality problems, such as *missing value visualization*, or *outlier detection.* This round was carried out without any time restriction.

As a result all the selected references from the overview phase were categorized into the following groups:

- Visually indicated
- Visually encoded
- Incorporated

2.1.3 Search Criteria

The following criteria were taken into consideration during the selection phase:

- **Timeliness** for each search engine only articles that weren't older than 10-15 years (after 2000) were taken into consideration.
- **Number of Cites** provided by Google Scholar and Microsoft Academic Search - as it to some extend reflects whether a given resource is well known and if it's relevant to a broad audience, or only to a minor set of people.
- **Document meta data** that answers the following questions:
  - **Title:** Do all keywords occur in the title of the result, and do they make sense in the respective setting?
  - **Abstract:** Are the terms mentioned in the context of visualizing/indicating data quality problems, or rather randomly? Is the contents similar or comparable to the problem at hand?
  - **Conclusion:** Are the results relevant to the questions of this work? Are the results transferable, and can they be applied to the problems of this work?
- In all cases only the results of the first five pages where taken into consideration when selecting resources.

2.1.4 Key Word Refinements

In order to refine keywords both, Google Trends and Google was used to compare two keywords by their (normalized) search volumes, as it reflects the importance and usage of it. This allowed to compare similar keywords (synonyms) and decide which one is more commonly used and appropriate for the search. For instance, instead of using the keywords *erroneous data* (2.5 million hits) one is more likely to find relevant content by using the keywords *invalid data* (40. million hits), as the latter is used more frequently.

2.2 Introduction to Data Quality and its Dimensions

Even today no single valid definition for the term data quality exists, as mentioned by Sadiq et al. [Sad13]. This absence of a common understanding of data quality can be partly explained by a user centric view of data quality. Nevertheless, various user centric definitions exist and try to give an universal answer to when data is considered to be of high quality. For instance, one such answer is provided by A. B. Godfrey [God99]:

> "Data are of high-quality if they are fit for use in their uses (by customers) in operations, decision-making, and planning. They are fit for use when they are free of defects and possess the features needed to complete the operation, make the decision, or complete the plan." [God99]

Also the author of Sadiq et al. [Sad13] emphasize the importance of the „customer“ and support the view that data quality is exclusively defined by the user.

> “Customer needs, at the moment of use, dictate what quality means and the moment of creation dictates whether customer needs will be met.” [Sad13]

The fact that data quality is hard to define is also underlined by Huang [Hua05], who claims that „to date there has been no uniform and rigorous definition of data quality“. He argues that such a definition would include many different dimensions, which should be introduced in any phase of a visualization, starting at the data acquisition. Additionally, over the last decade several methods were proposed to convey data quality. One theoretical foundation that is often cited by data quality visualization literature, is Pipino et al. [PLW02], which comprehensively describes data quality dimensions. An important conclusion from this work is that “*one size fits all set of metrics is not a solution*”, emphasizing the individuality when thinking about data, quality, visualization, and decision-making.

This attitude towards data quality dimensions is also underlined by Sadiq et al. [Sad13], where the authors conclude that data quality dimensions are strongly dependent on the respective organization and their problems. In their work they refer to Levitin et al. [LR93] and Wang et al. [WS96], two essential starting points, on which one can base his or her own dimensions of data quality. At last, I found the taxonomy of Gschwandtner et al. [GGAM12], which proposes the following three data quality dimensions that were later used throughout the design and implementation:

- **Missing Data** (missing values, missing timestamps)
- **Implausible Values** (outliers)
- **Wrong Data** (invalid Values)

At last, I sum up data quality by using the following quote: „Data Quality and Uncertainty Visualization are like the weather, everyone’ talks about it, but no one does anything about it.[Dem06]“

2.3 An Overview of Data Quality Visualizations

In this section data quality visualization solutions are examined in more detail. These visualizations were categorized into one of the following three groups in order to ensure comparability:

- **Data quality, visually indicated:** contains all approaches that only indicate data quality issues on the raw (tabular) data, rather than within the actual visualization of the data.

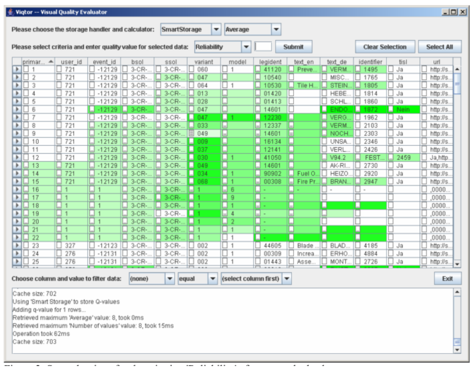
- **Data quality, visually encoded:** includes all approaches where data quality information is visualized within the actual data visualization.

- **Data quality, incorporated as a new dimension:** holds visualizations that introduce data quality as a new dimension, linked to interactive views.

2.3.1 Data quality visually indicated

VIQTOR, is a tool that tries to gather data quality ratings directly from the user, while tackling problems such as *multiple criteria*, *subjectivity*, *multiple users*, *granularity*, as well as *sporadic scores*. Führing and Naumann [FN07] argue that the users experience, regarding the knowledge about data quality, „cannot be transferred into a comprehensive and common rule". Therefore, VIQTOR represents a solution that allows the user to enter (i.e., rate) and display several different data quality criteria at the same time (see Figure 2.1a). In this context the authors list quality indicators such as reliability, completeness and believability. By using VIQTOR all of these quality criterion can be rated, aggregated and displayed. Additionally, they examine possible ranges (i.e., numerical, categorical, etc.) for these indicators and discuss problems that might be encountered when aggregating them. For instance, the arithmetic average for 2 and 200 users might be the same, however, without additional information, the expressiveness is not.

DAVIS, an approach proposed by Sulo et al. [SEG05], visually indicates data defects on raw data in a tabular form (see Figure 2.1b), but does not incorporate the visualization into the data exploration process. Nevertheless, their approach demonstrates the utility of visualization and their contribution shows that it leads to faster understanding of the data and better decision-making.



Please choose the storage handler and calculator: SmartStorage Average

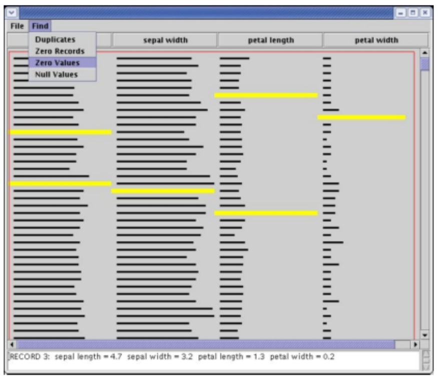
Please select criteria and enter quality value for selected data: Reliability

primar...	user_id	event_id	bsol	ssol	variant	model	legident	text_en	text_de	identifier	tisl	url
1	721	-12129	3-CR-...	3-CR-...	060	1	41120	Preve...	VERM...	1495	Ja	http://s...
2	721	-12129	3-CR-...	3-CR-...	047		10540		MISC...	1765	Ja	http://s...
3	721	-12129	3-CR-...	3-CR-...	064	1	10530	Tile H...	STEIN...	1805	Ja	http://s...
4	721	-12129	3-CR-...	3-CR-...	013		01420		HEBE...	1814	Ja	http://s...
5	721	-12129	3-CR-...	3-CR-...	028		01413		SCHL...	1860	Ja	http://s...
6	721	-12129	3-CR-...	3-CR-...	047		14601		ENDO...	1872	Nein	http://s...
7	721	-12129	3-CR-...	3-CR-...	047	1	12230		VERG...	1962	Ja	http://s...
8	721	-12129	3-CR-...	3-CR-...	033		12337		VERM...	2103	Ja	http://s...
9	721	-12129	3-CR-...	3-CR-...	049		14601		NOCH...	2303	Ja	http://s...
10	721	-12129	3-CR-...	3-CR-...	009		16134		UNSA...	2346	Ja	http://s...
11	721	-12129	3-CR-...	3-CR-...	037		12141		VERL...	2426	Ja	http://s...
12	721	-12129	3-CR-...	3-CR-...	030	1	41050		V94.2	FEST...	2459	Ja,http...
13	721	-12129	3-CR-...	3-CR-...	049		14601		AK-RI...	2730	Ja	http://s...
14	721	-12129	3-CR-...	3-CR-...	034	1	90902	Fuel O...	HEIZO...	2920	Ja	http://s...
15	721	-12129	3-CR-...	3-CR-...	068	1	00308	Fire Pr...	BRAN...	2947	Ja	http://s...
16	1	1	3-CR-...	3-CR-...	1	6	-	-	-	-		,0000...
17	1	1	3-CR-...	3-CR-...	1	9	-					,0000...
18	1	1	3-CR-...	3-CR-...	1	1	-					,0000...
19	1	1	3-CR-...	3-CR-...	1	4	-					,0000...
20	1	1	3-CR-...	3-CR-...	1	2	-					,0000...
21	1	1	3-CR-...	3-CR-...	1	1	-					,0000...
22	1	1	3-CR-...	3-CR-...	1	1	-					,0000...
23	327	-12123	3-CR-...	3-CR-...	002	1	44605	Blade ...	BLAD...	4185	Ja	http://s...
24	276	-12131	3-CR-...	3-CR-...	002	1	00309	Increa...	ERHO...	4884	Ja	http://s...
25	276	-12131	3-CR-...	3-CR-...	002	1	01443	Asse...	MONT...	2726	Ja	http://s...

Choose column and value to filter data: (none) equal (select column first)

```
Cache size: 702
Using 'Smart Storage' to store Q-values
Adding q-value for 1 rows...
Retrieved maximum 'Average' value: 8, took 0ms
Retrieved maximum 'Number of values' value: 8, took 15ms
Operation took 62ms
Cache size: 703
```

(a) VIQTOR - a tool to rate and display data quality [FN07].



RECORD 3: sepal length = 4.7 sepal width = 3.2 petal length = 1.3 petal width = 0.2

(b) The DAVIS visualization tool[SEG05].

Figure 2.1: Solutions that visually indicate data on tabular data.

A more recent solution is *Profiler*, which supposedly is an "*extensible system for data quality assessment*" (Kandel et al. [KPP+12]). While this tool already provides several dynamic summary views, which aid the detection of data quality problems, it lacks the support for time-oriented data. Figure 2.2 shows these different views. In contrast, one novel aspect of *Profiler* is that the user is offered with data cleansing (i.e adjustment of data values) functionalities in order to improve data quality. In addition, *Profiler* already makes use of different views, which later can also be seen in the third group of data quality visualization approaches. The reason why this solution does not fall into this third group is because it lacks the incorporation of data quality into the data exploration process.

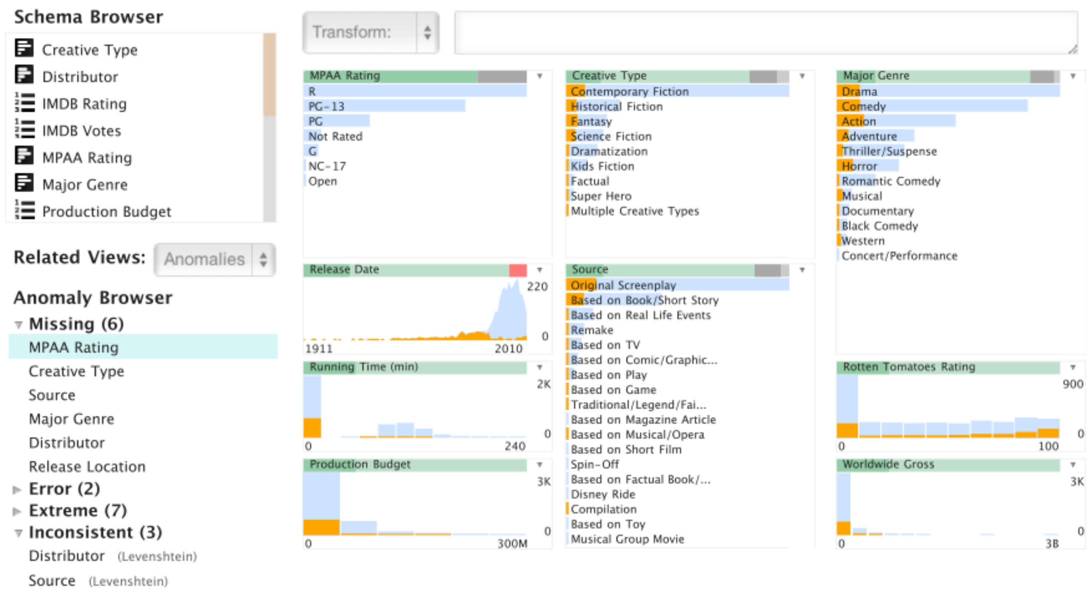


Figure 2.2: A screenshot of the tool Profiler [KPP+12].

2.3.2 Data quality visually encoded

In contrast to the first group, Xie et al. [XHWR06, ZWRS07] as well as Rundensteiner et al. [RWX+07] incorporate data quality into the exploration process for multivariate data. Most of their findings are based on implementations in the tool XmdvTool (see Figure 2.3, 2.4 and 2.5)[1]. In this context Xie et al.'s [XHWR06] two major contributions are a framework to define data quality measures as well as approaches to visualize data quality. Their first approach, called *new dimension*, treats data quality as an additional dimension that is added to the original data set. For instance, in parallel coordinates this new dimension would be represented by an additional axis. The second approach, called *visual encoding*, maps data quality measures to the original visualization of the data [XHWR06] (see Figure 2.4).

[1]available: `http://davis.wpi.edu/xmdv/` (retrieved June 16, 2014)

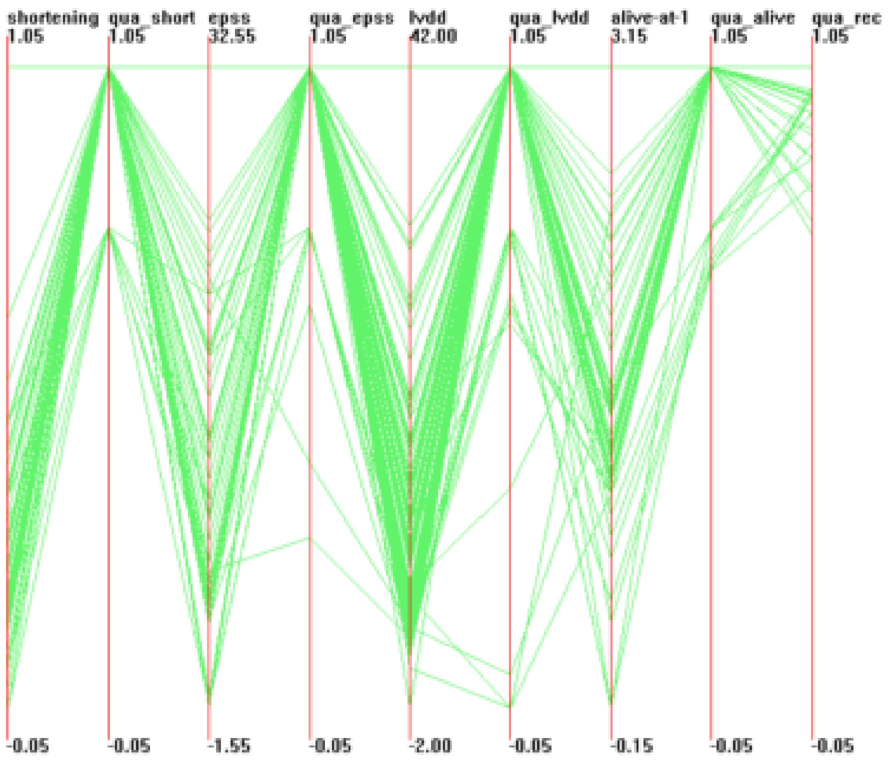


Figure 2.3: XmdvTool - quality information as new dimension [RWX+07].

In this context the master thesis of Huang [Hua05] shares most of the conclusions with early papers of Xie and Rundensteiner et al., but provides more detail about the actual implementation. For instance, the thesis discusses different imputation algorithms (*nearest neighbor* and *multiple imputation* used to acquire data quality values. An important contribution of Huang's work is that using the third dimension proved to be a poor choice when visualizing data quality.

Another example of data quality visualization is a poster paper of Tekušová et al. [TKSK08], which discusses three different ways (see Figure 2.6) of incorporating data quality information. First, they propose *multivariate glyphs* mapped to a color saturation stripe legend, which offers the possibility to show glyphs for every data point. However, putting glyphs too close to each other might cause occlusion. According to Tekušová et al., this occlusion can be overcome by using interaction techniques such as rotate or zoom-in. Furthermore, they use *transparency* as another way of displaying data quality information, by mapping the level of uncertainty to the level of transparency. The third way they propose in their paper is *texture overlays* to encode data quality information of either qualitative (predefined textures), or quantitative information (texture patterns - i.e. color, frequency or direction). This poster paper is an interesting thought-provoking

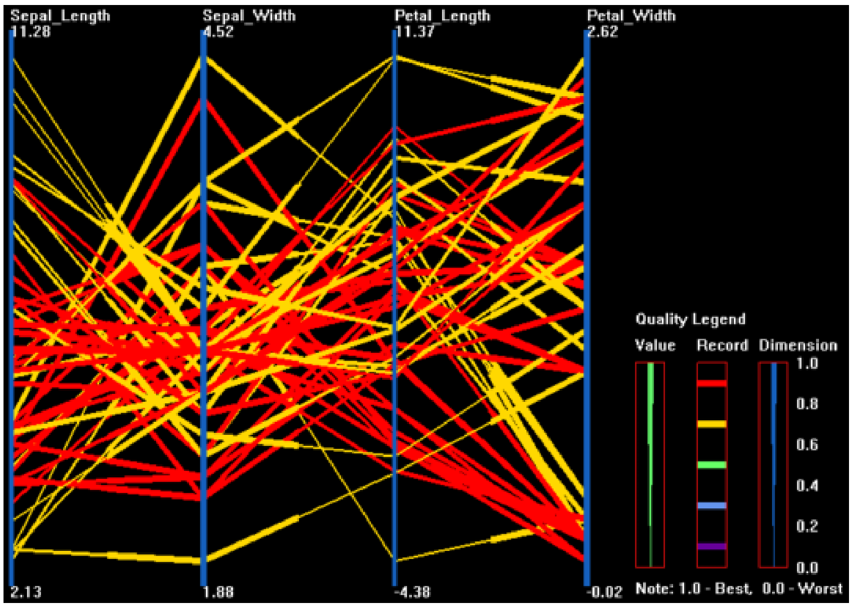


Figure 2.4: XmdvTool - quality information encoded into a parallel coordinate diagram [RWX+07].

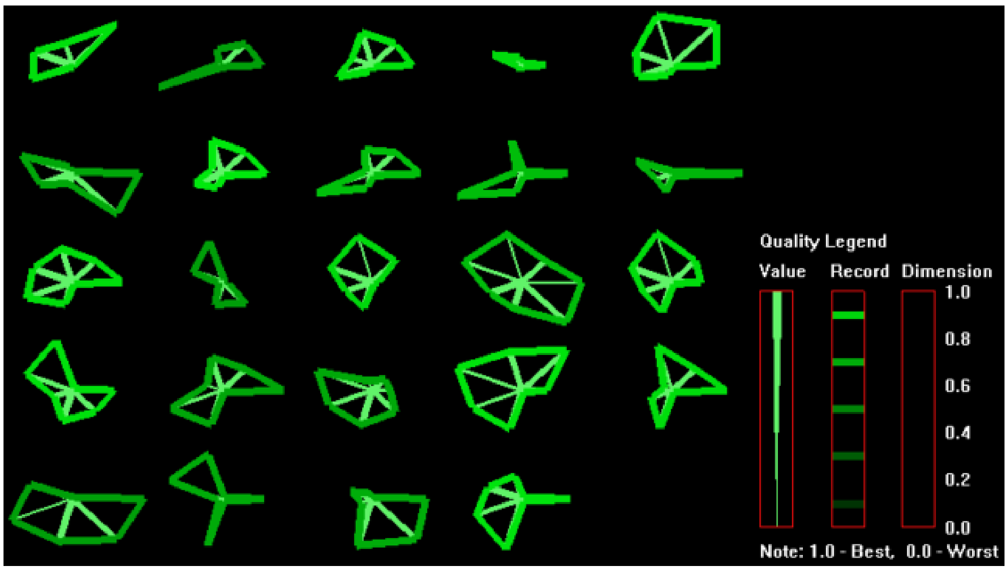
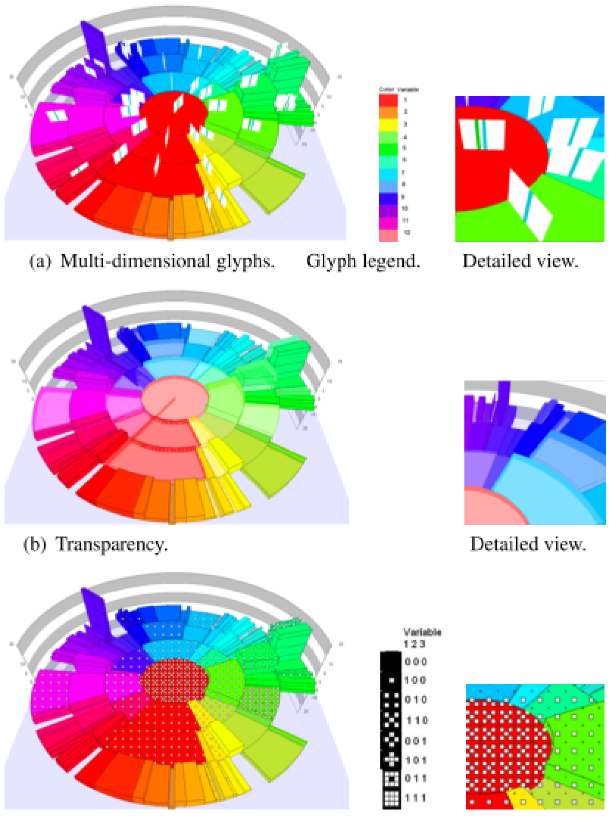


Figure 2.5: XmdvTool - quality information encoded into a star glyph [RWX+07].



(a) Multi-dimensional glyphs. Glyph legend. Detailed view.

(b) Transparency. Detailed view.

(c) Texture overlay. Texture legend. Detailed view.

Figure 2.6: 3 ways of displaying quality information within the visualization [TKSK08].

impulse. However, as also mentioned by the authors itself, their work lacks a proper evaluation based on a user study, which makes it less effective as a guideline.

In parallel to the work of Xie et al. [ZWRS07] and Rundensteiner et al. [RWX^{+}07], Shankaranarayanan and Zhu [ZSC07] investigated how meta dat a on data quality impacts decision performance, emphasizing the need to provide decision makers with such meta data. In their work they propose a prototypical tool named *SPIDEV*, where data quality is visually encoded into a spoke chart. In a latter work Shankaranarayanan and Zhu [SZ12] examined the effectiveness of their approach by means of controlled experiments (see Figure 2.7). In these experiments, they demonstrate that additional quality meta data lowers the perceived mental demand, improves the decision accuracy, leads to shorter decision times, and increases the confidence in the decision itself. An interesting question the author raise is whether the visualization of data quality leads to cognitive overload. In that context they found that when a task is of low complexity, the incorporation of quality meta data negatively impacts the decision outcome. One proposed solution for this is to find and use a proper visualization technique in order to reduce the cognitive overload involved.

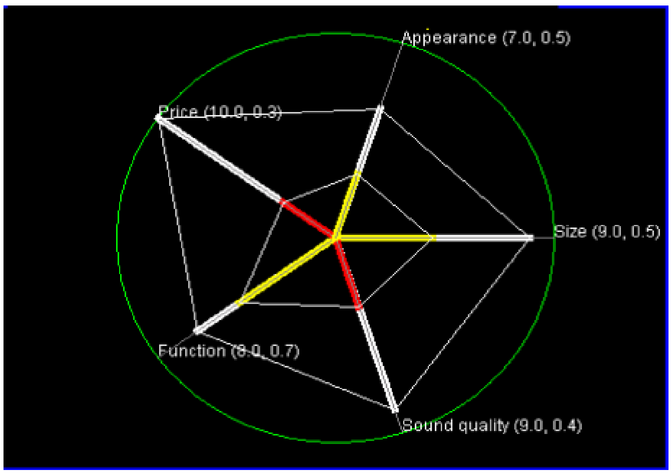


Figure 2.7: SPIDEV - Metaphor for Visualizing Data and QM [SZ12].

2.3.3 Data quality incorporated as a new dimension

One of the early approaches to visualize data quality of multivariate, time-oriented data is the contribution of Vlag et al. [VdVK04]. In their paper they visualize data quality by using a *temporal ordered space matrix* (see Figure 2.8), which is added as an interactive view to the actual data exploration process. They particularly make use of the principle of brushing and linking, which provides the user with multiple different views at the same time. In their solution linkage guarantees that when the user changes one of the provided views, all the other views are updated accordingly. For example, when the user uses the temporal ordered space matrix, he or she can set a threshold for the data quality, which automatically displays only matching data in the other views. Ultimately, their approach facilitates the use of interaction, as well it promotes the incorporation of data quality *interactively* into the data exploration process.

A similar approach is introduced by Xie et al. [ZWRS07], which extends on previous work of Xie et al. [XHWR06], by adding the concepts of *data space* and *quality space* (see Figure 2.9b). These two concepts essentially divide the visualization into two parts, the actual visualization of the data (i.e., the data space) and the visualization of data quality (i.e., the quality space). The latter can also be multiple smaller views that are linked to the data space. Additionally, they suggest the concept of *quality maps* as a new way of visualizing the quality space. For this, they provide two different types, namely Stripe Quality Maps (see Figure 2.9a) and Histogram Quality Maps (see Figure 2.9b). Those interactive brushing and linking principles allow the user to explore the data space by interacting with those two types of quality maps, where the linkage between the data and quality space works in both directions.

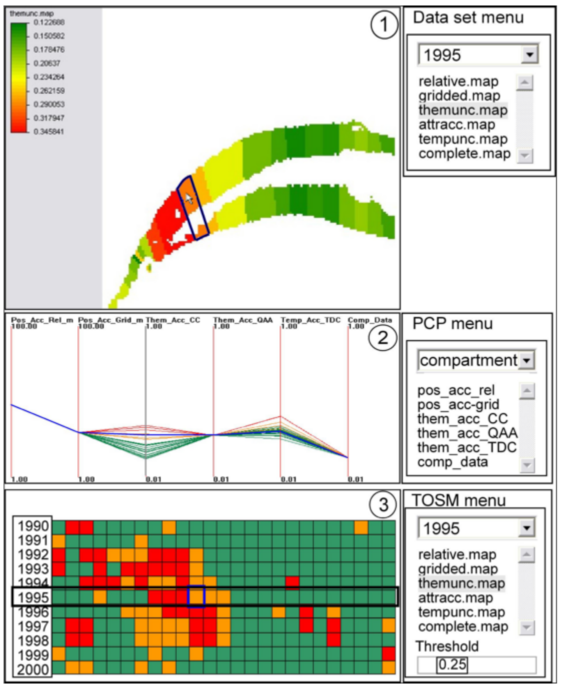
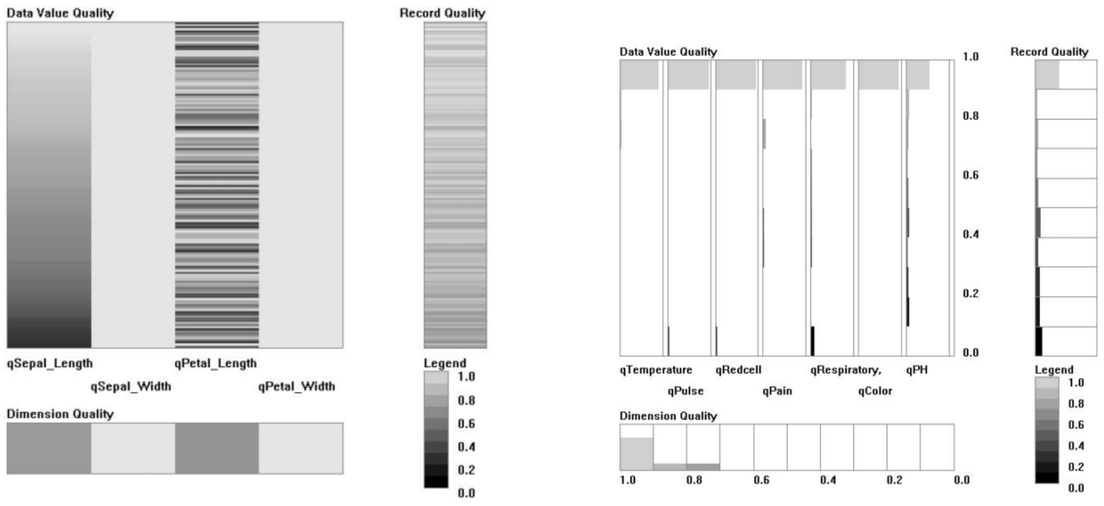


Figure 2.8: Temporal ordered space matrix [VdVK04].



(a) Stripe Quality Map [ZWRS07]. (b) Histogram Quality Map [ZWRS07].

Figure 2.9: Different quality maps for visualizing quality space.

Huang [Hua05], who introduced quality sliders in his thesis (an early form of quality maps), emphasizes on the need to display data quality separately (i.e. with *quality maps*) in order to allow interaction with existing techniques such as brushing and linking. In their work they propose several different quality sliders that are linked with a parallel coordinates plot, where changes of the quality sliders directly affect the plot. Figure 2.10 shows these separately linked views.

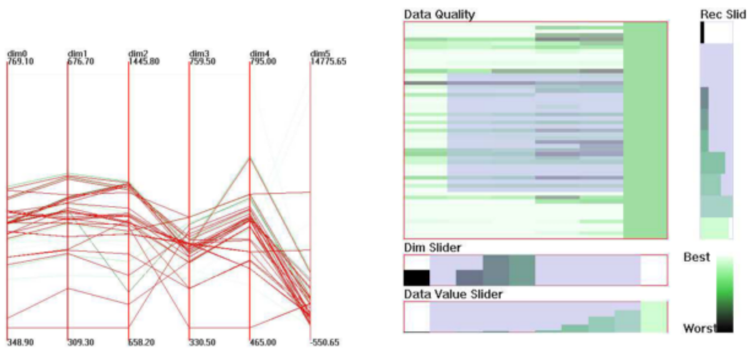


Figure 2.10: Data Quality Visualization with separate linked views [Hua05].

In a more recent work, Rundensteiner et al. [RWX$^+$07] emphasize the need to make the whole data exploration process a quality aware process. This means to address the data quality also during data transformation, query processing and visual mapping. Furthermore, they identify visual clutter as a visual mapping quality problem that might even obscure the actual structure, as well as they show how such issues can be addressed. For instance, they demonstrate an algorithm that reorders the axes of a parallel coordinate visualization in order to reduce clutter. In a latter work Ward et al. [WXYR11] introduces further measures such as *abstraction quality* (see Figure 2.11) to further integrate data quality into the visualization process, while similar ideas were already suggested in Cui et al. [CWRY06]. In their proposed solution the users can interact with the data and adjust the *data abstraction level*, while seeing the results. These results are then shown in a *histogram difference measure*, as well as in a *nearest neighbor measure* (see Figure 2.12).

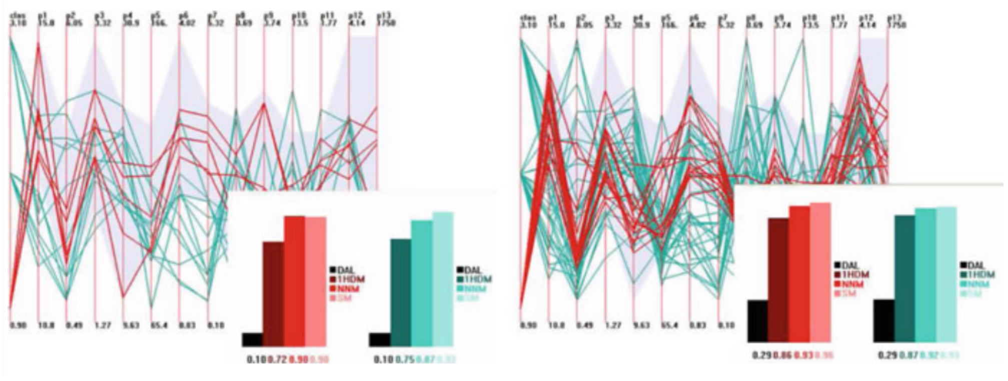


Figure 2.11: Missing values encoded into the visualization process [WXYR11].

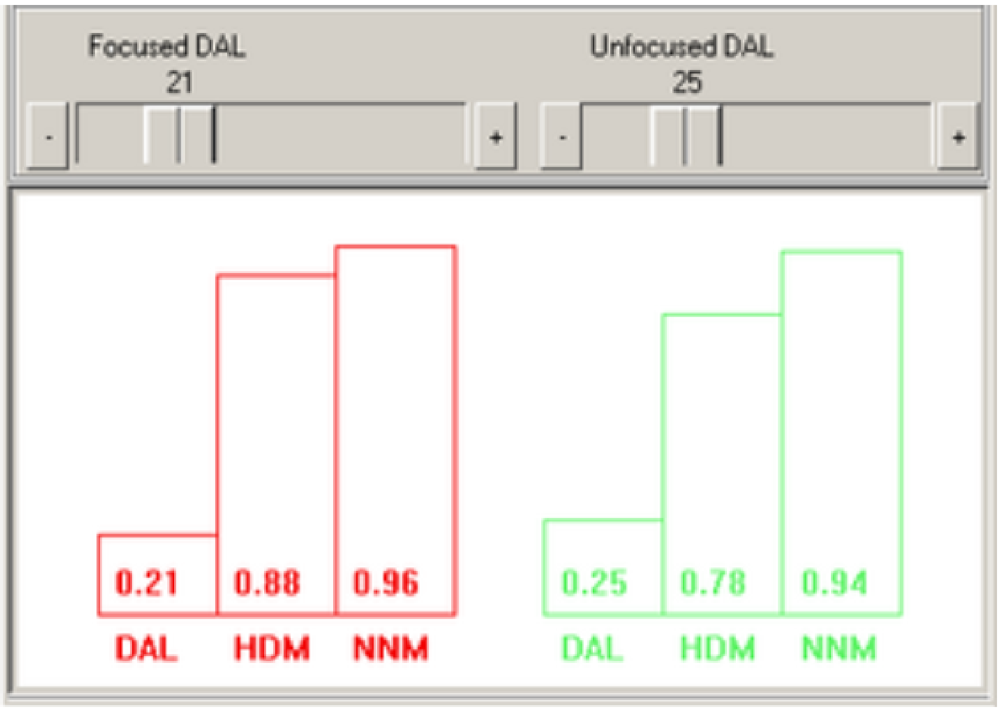


Figure 2.12: Histogram difference measures for missing values [CWRY06].

More importantly, in [WXYR11] they conclude that *quality maps* are preferable to visually encoding data quality, as the latter causes cognitive overload and increases the complexity of decision-making tasks in case of a simple task. Table 2.1 gives an overview of all the different data quality visualization approaches examined.

Table 2.1: Overview of data quality visualizations in combination with multivariate time-oriented data.

Approach	**Year**	**Type**	**Mult.**	**Time.**
Visually Indicated				
DAVIS [SEG05]	2005	Table	-	-
VIQTOR [FN07]	2007	Table	-	-
Profiler [KPP+12]	2012	Table, Stacked Histograms, Area Charts	-	-
Visually Encoded				
3D Integration - Huang [Hua05]	2005	Parallel Coordinates	✓	-
New Dimension - Xie et al. [XHWR06]	2006	Parallel Coordinates, Scatter plot matrice	✓	-
Visually Encoded - Xie et al. [XHWR06]	2006	Parallel Coordinates, Scatter plot matrice, Star Glyph Plot	✓	
Visually Encoded in hierarchical data - Tekušová et al. [TKSK08]	2008	3D Circular View	✓	-
Improves Decisions - Shankaranarayanan and Zhu [SZ12]	2012	Spoke Wheel	-	-
As interactive View				
Temporal Ordered Space Matrix - Vlag et al. [VdVK04]	2004	Parallel Coordinates, Temporal Ordered Space Matrix	✓	✓
Quality Sliders - Huang [Hua05]	2005	Parallel Coordinates, Quality Sliders	✓	-
Abstraction Quality - Cui et al. [CWRY06]	2006	Parallel Coordinates, Scatter plot matrice	✓	-
Quality Space - Xie et al. [ZWRS07]	2007	Parallel Coordinates, Quality Maps	✓	-
Visual Quality - Rundensteiner et al. [RWX+07]	2007	Parallel Coordinates	✓	-
Abstraction Quality - Ward et al. [WXYR11]	2011	Parallel Coordinates, Cluster Hierarchy, Star Glyphs, Cityscape, Quality Maps	✓	-

2.4 Specific Solutions

2.4.1 Missing Values

Missing values are one of the most prevalent problems within the data pre-processing step, as well as one of the three major data quality problems identified in the problem statement (see Section 1.1). A common approach to deal with such missing values is to use deletion. Two most common forms are listwise and pairwise deletion [End10].

Listwise deletion, also known as complete case analysis, leaves out all records (cases) that contain missing values, which is especially effective when the provided data set contains only a few missing values [Hua05]. In contrast, pairwise deletion, also known as available case analysis, only removes variables with missing values only for single computed statistics. However, this way all the different computed statistics differ depending on the variables of interest and as a result can not be compared [IBM14]. Ultimately, both alternatives have disadvantages such as loss of precision and statistical power. Moreover, the following assumption applies to both:

> "Note that both LISTWISE and PAIRWISE deletion methods make very strict assumptions about the mechanisms that cause data to be missing. In order for these methods to produce appropriate results in most situations, data must be what is known as MCAR, or missing completely at random, meaning that the missing values must be unrelated to the observed values" [IBM14]

As a result, using deletion in combination with data quality visualization hides the data's imperfection from the analyst and might even mislead during the decision-making process.

Further options are single imputation methods, or model based methods. The former makes use of algorithms such as mean/mode substitutions, dummy variables, or single regression, whereas the latter uses maximum likelihood, multiple imputation, or nearest neighbor algorithms. All these methods and algorithms share one common goal, which is to fill in the missing value gaps [Hua05, TAF12]. However, resources such as Huang [Hua05] conclude that these methods cannot be used to tackle all missing value cases, as they strongly depend on the underlying assumptions. Thus, Huang proposes to use visualization in order to pick the right algorithm and to further refine them. In this respect Huang [Hua05] and Templ et al. [TAF12] mention XGOBI (see Figure 2.13) and MANET (see Figure 2.14). Two tools that deal with missing values by using *statistical inference methods* such as multiple imputation, which make use of information visualization in order to further adjust the algorithm finally used.

Wang and Wang [WW07], in contrast to all the methods mentioned so far, encourage to use a clustering technique called Self Organizing Map (SOM), an unsupervised learning algorithm, in order to visualize and detect missing value patterns. In their work they use the SOM to map high-dimensional data onto low-dimensional pictures, which then can be used to detect patterns of missing values. According to the authors, one of the advantages of SOM over statistical methods is that the former can be easily applied on high-dimensional data, whereas the latter might lead to biased analysis results and invalid conclusions. Furthermore, the authors argue that using a simple measure such as the number of incomplete entities over the number of total entities not just ignores the existence of patterns (see Figure 2.15), but also the fact that real world data often does not have a regular multivariate distributions. The result of this is a SOM as a

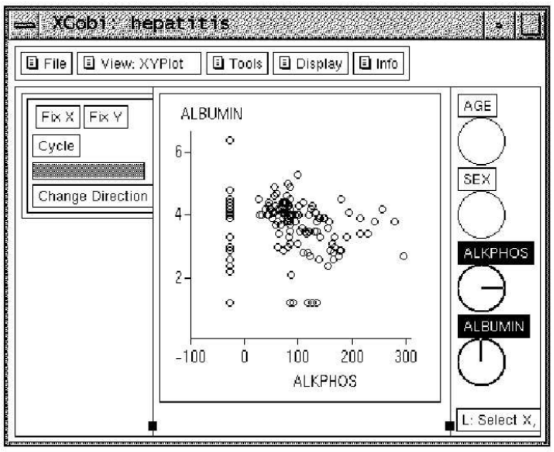


Figure 2.13: XGOBI scatter plot [TAF12].

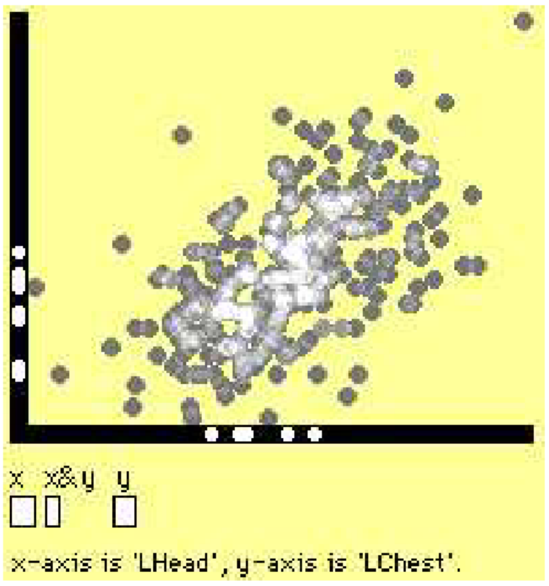


Figure 2.14: MANET scatter plot [UHHS96].

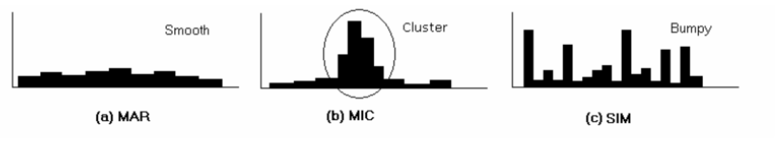


Figure 2.15: Different patterns regarding missing values [WW07].

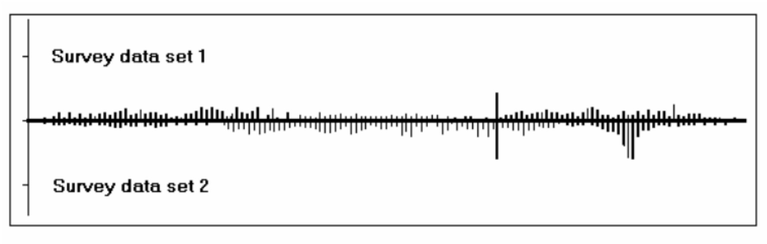


Figure 2.16: Result of the approach of Wang and Wang [WW07].

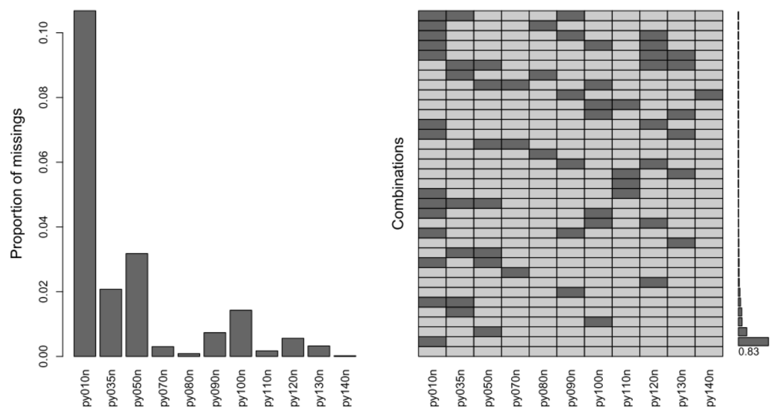


Figure 2.17: Aggregation Plot by Templ et al. [SCB98].

one-dimensional graph, similar to a histogram, which can be used to detect patterns (see Figure 2.16).

Templ et al. [TAF12] propose *VIM*, a tool that provides „an overview of the amount of missing values and to detect monotone missing values patterns“. For instance, the authors propose an *aggregation plot* (see Figure 2.17), which shows the number of missing values for each variable in the data. Another interesting plot is the *matrix plot*, as it is very similar to the quality bands (stripes) introduced by Huang [Hua05] (see Figure 2.18).

Although VIM offers a variety of different options to display data quality, it does not provide intuitive interactions in combination with the various plots.

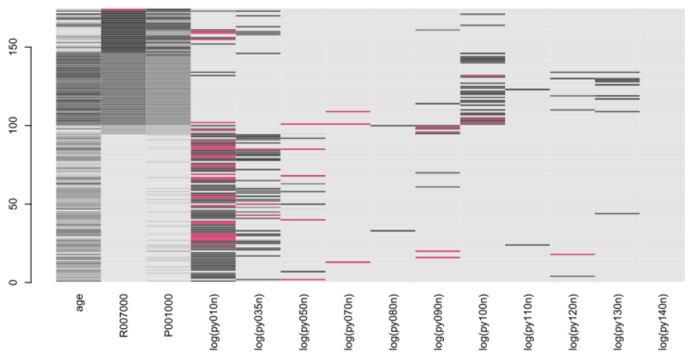


Figure 2.18: Matrix plot by Templ et al. [UHHS96].

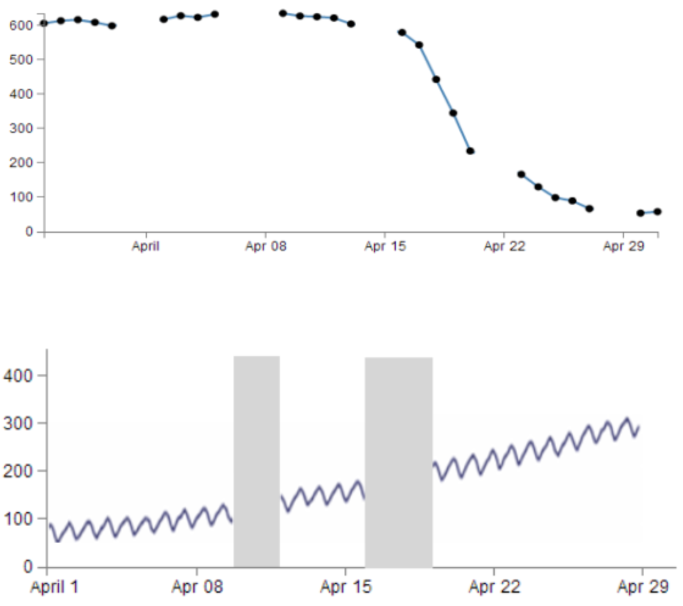


Figure 2.19: Different options to encode missing values into the visualization process.

In slides provided by the partner of this work three approaches to display missing values were suggested. The first is to use gaps in a line plot and the second to display spikes/bars at the position where the missing values occur (see Figure 2.19). The third approach is to use a dotted line whenever missing values occur (see Figure 2.20). Furthermore, an overview bar (quality stripe) is introduced as an extension that is positioned on top of the line plot to show in overview where missing values occur.

Eaton et al. [EPD05] cover another important aspect of missing values, namely, *how users interpret graphs with missing data*. In their study they suggest that „user may not

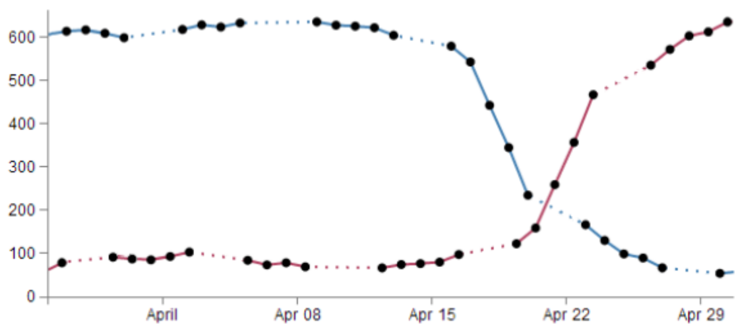


Figure 2.20: Missing values displayed as dotted lines in 2 variables.

realize that data is missing when it is replaced by a default value“. Furthermore, they examine that users tend to make „general conclusions“ with partial data. Their study involves three different displays. In the first they use a misleading display, where all missing values are replaced by a default value (0). In the second, they completely omit (absent) missing data, while in the third they omit missing values, but add additional indicators to each missing data point. As a result, the participants of this study prefer the third display (i.e., the coded display), where missing values are indicated by visual elements.

2.4.2 Invalid Values

Another identified data quality problem is invalid values (also refereed to as erroneous data). In order to identify invalid values one has to formulate logical constraints that define the semantics of the data. For instance, one such constraint would be *male* or *female* as the only valid values for a gender field. The authors of Sadiq et al. [Sad13] describe the confidence of such a constraint as a measure of how much percent of the data satisfy a given constraint. Moreover, they mention that sensor data might be prone to systematic or correlated problems. And propose to identify subsets of the data that satisfy or violate a constraint. Finally, they advocate statistical error detection techniques (probabilistic) as an extension to constraint-based error detection (satisfying/failing).

2.4.3 Outliers

Outliers, abnormalities, extremes, or anomalies are all terms that describe a situation where some data values differentiate substantially from the rest of the population. Outliers may provide evidence that something in the data generating process went wrong, which information can then be used for fraud or intrusion detection, process improvements, and many more. Achtert et al. [AKR$^+$10] describes outliers as follows:

> “Outlier detection can be seen as not merely interested in removing noise but also in finding interesting database objects deviating in their behavior

considerably from the majority and, as such, providing new insights"

A good introduction to outlier detection is the paper of Hodge et al. [HA04], which introduces a survey of contemporary techniques for identifying outliers. Another very detailed resource in this field is a book written by Charu C. Aggarwal's [Agg13], which contains several approaches such as *probalistical/statistical models*, *linear models*, or *proximity based models* in order to deal with outliers. In his work the author examines several advantages and disadvantages of these different techniques. For instance, he indicates that statistical models have the advantage of being easily applicable to all kinds of data sets. Furthermore, he advocates the importance of multi dimensional models, as he believes them to be great input to future algorithms, even though they are computationally expensive. On the other hand, Achtert et al. [AKR+10] argues that all those "classical" methods do actually not decide on whether a data point is an outlier, but only give a score or factor of its *outlierness*. Those scores and factors often *„widely differ in range, contrast and expressiveness"*, and underline the method's scaling issues. For instance, while some methods assign high scores to certain data points, others do the complete opposite and by that indicate outliers as normal data. Thus, the authors conclude that all the methods depend on the scenario as well as the data used. Moreover, the authors suggest that this is one of the reasons that makes it especially hard for novices to interpret such scores. As a response, the authors propose a tool that visualizes such scores and factors for high dimensional data. This tool implements a range of different detection methods and is based on a scatter plot. Outliers in this tool are indicated by dark red bubbles, where the magnitude of the score is visually encoded in the radius of the bubble. Furthermore, the tool allows to visualize either all outliers with different methods, or only the top-k outliers, while also providing the possibility to zoom-in and zoom-out, and only look at specific areas. Although the authors propose a new way of visualizing outliers, they do not provide the reader with any evaluation, pointing out advantages and disadvantages of the solution.

One classical way to find, but also to visualize outliers, is to use the well known J. W. Tukey Box-Whisker Plot. Outliers either lie in the 3x Interquartile Range (IQR), after the outer fence, or can be suspected within the 1.5 IQR (see Figure 2.21), after the inner fence. On his site Pandre[Pan14] lists a few different ways to visualize outliers, which ranges from scatter plots to radar charts and parallel box-whisker plots.

A statistical model approach would be a *funnel plot*, as suggested by Ieva et al. [IP15] (see Figure 2.22). First, they introduce a funnel plot as a technique to visualize outliers. Then they consider random intercept methods as an additional outlier detection technique and finally propose to use both approaches to easily find and visualize outliers.

In Kim et al. [KK07] the authors introduce a visualization of clustering algorithms for identifying multiple outliers. First, they review different clustering algorithms such as single linkage clustering, originally proposed by Sebert et al. [SMR98]. They then use the results of the clustering and combine it with a minimal spanning tree (MST) to visualize the clustering process, as well as to examine the model's adequacy (see Figure 2.23). As a

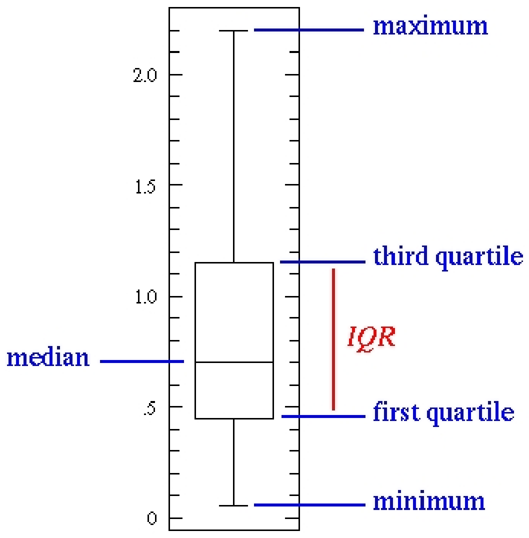


Figure 2.21: A visual explanation of IQR [Pan11].

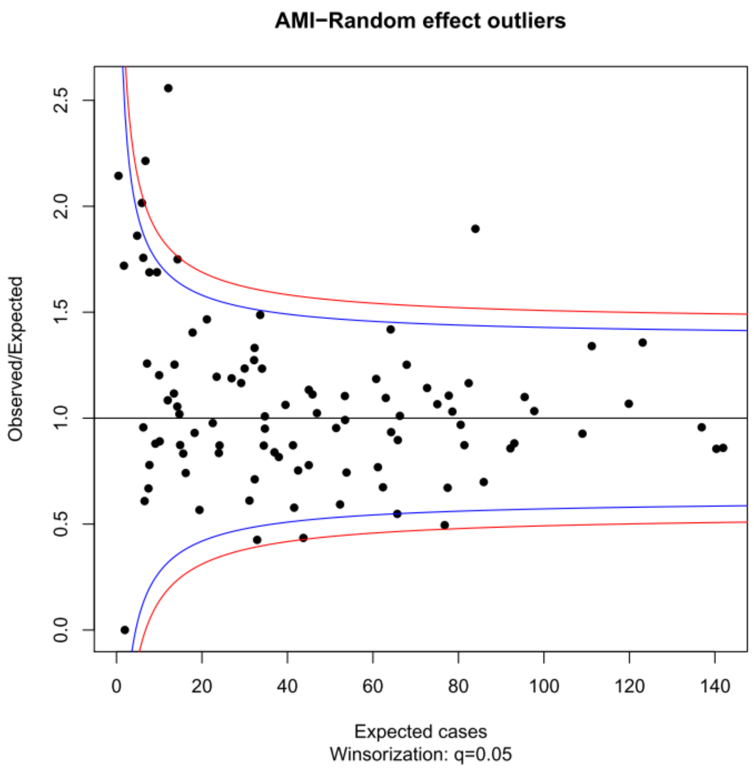


Figure 2.22: A funnel plot, an option to display outliers [IP15].

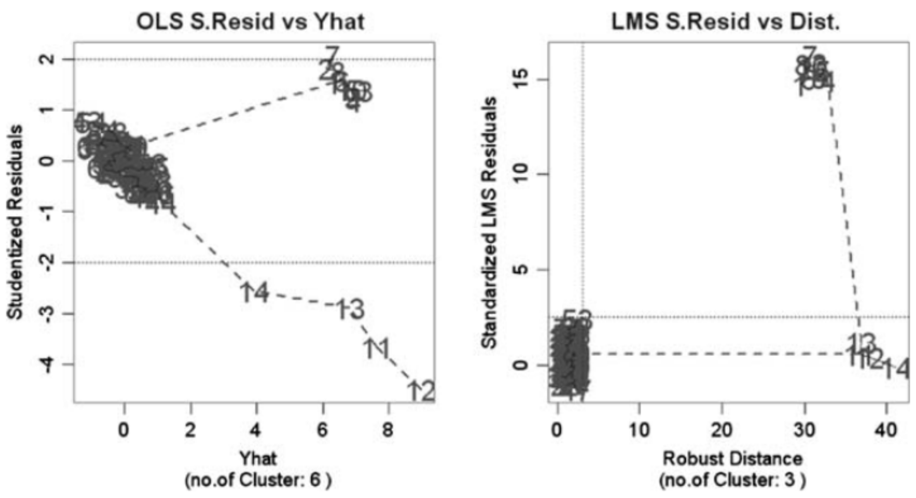


Figure 2.23: Outlier preserving focus+context visualizations [SMR98].

result, they infer that some of their suggested approaches have a tendency towards either *masking* (detection of too few outliers) or *swamping* (detection of too many outliers) and for this reason propose a modified algorithm that reduces at least the swamping. Finally, the authors conclude that their approach is useful in a 2-dimensional space and notice that multi-dimensional graphical visualization will be subject to further work.

Liu et al. [LCW02] emphasize that outliers are often considered as mistakes rather than phenomena of interest and propose two approaches in order to deal with them. The first approach is „accommodation“, where statistical methods are used to estimate the robustness of the data against outliers. However, the main concern of this approach is the data, while outliers are secondary. Conversely, the second approach aims to identify outliers and decides whether outliers are retained or rejected. As the focus of their work lies on the latter approach, the authors also propose to use a SOM (see Figure 2.24) in order to distinguish between noisy outliers (i.e., mistakes) and noise-free outliers (i.e., phenomenons). In their work they introduce a solution to outliers based on domain knowledge, where only outliers that are likely to be noise are removed. The authors label their solution as a useful extension to existing techniques. The results are depicted in Figure 2.25.

In their work, Novotny and Hauser [NH06] look into scalability issues and outlier visualization of multivariate large data sets. Their main focus lies on not smoothing out information (i.e., outliers), while aggregating/interpolating data. Therefore, they propose an output-oriented method that uses focus+context (e.g; layers, alpha blendings, etc.) techniques applied on a parallel coordinate diagram. For performing the aggregation, Novotny and Hauser make use of data binning in order to further aggregate the data (see Figure 2.26).

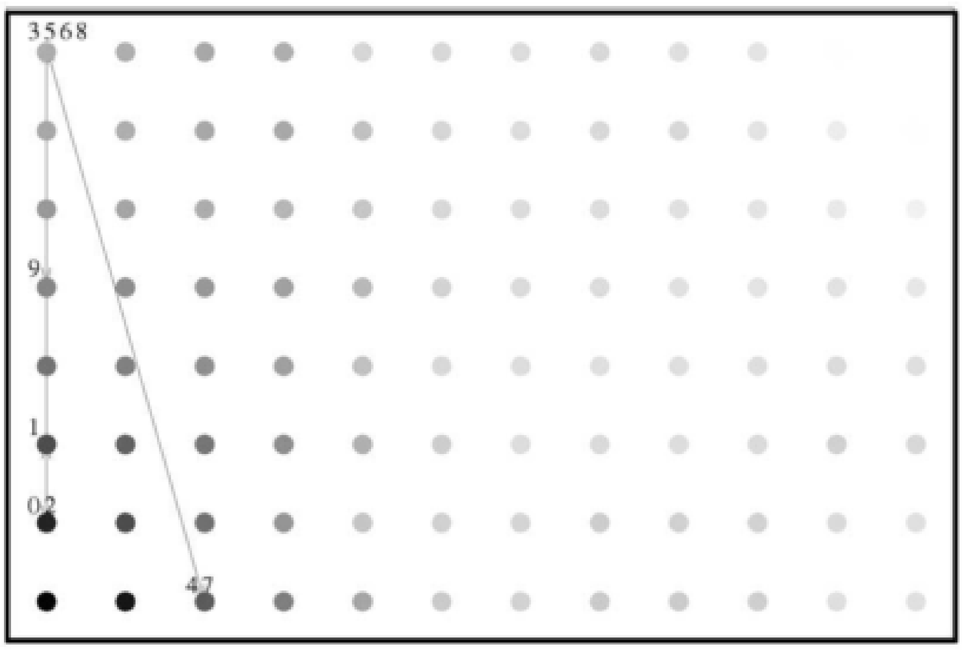


Figure 2.24: SOM: "*A glaucoma case (with cluster)*" [LCW02].

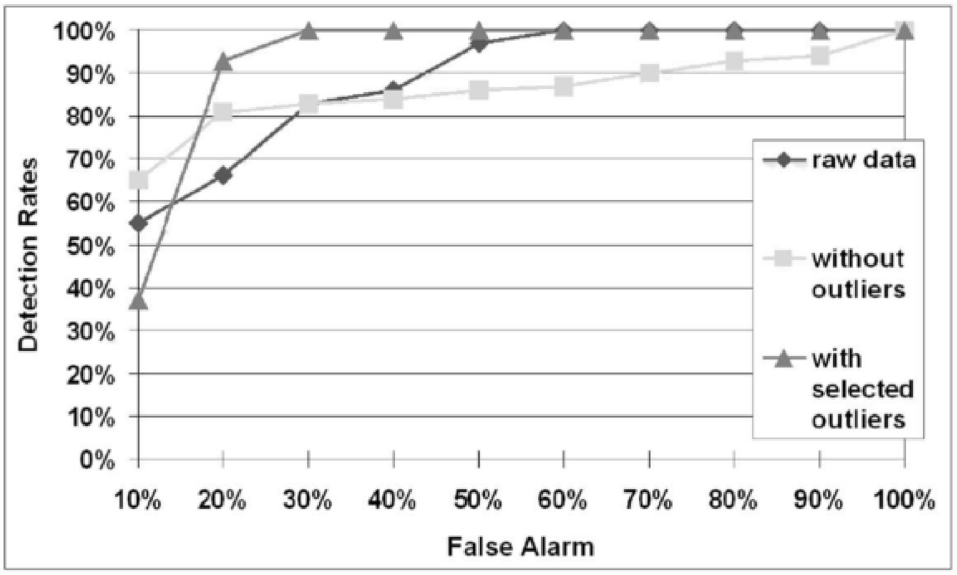


Figure 2.25: SOM: „*The results*“ [LCW02]

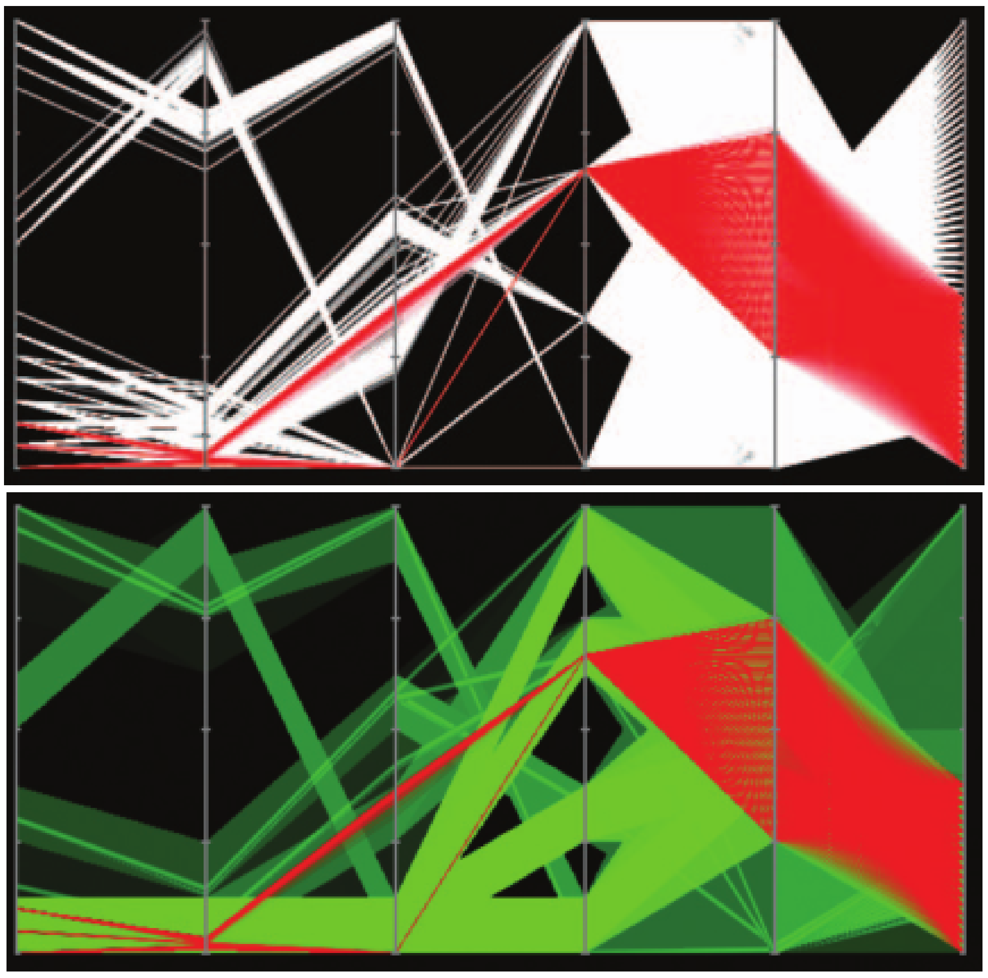


Figure 2.26: Data Binning to aggregate large data sets [NH06].

After the binning operation they apply different filters (e.g., isolation and median filter) with different thresholds in order to detect bins with visual outliers. These outliers are then extracted and visualized on a separate layer as polylines. According to the authors, this separation leads to a „more compact representation of major trends in the data“ and prevents „outliers from getting lost inside the context“. At last, the authors encourage to use density-based data binning, as the visualization performance is not dependent on the input data and that future challenges lie in the improvement of the outliers detection process, as well as the exploitation of modern graphics hardware.

Table 2.2 gives an overview of all the specific data quality problems identified:

Table 2.2: Overview of general data quality visualizations.

Method	**Year**	**Type**
Missing Values - Algorithms		
Deletion [End10]	2010	Complete Case, Available Case
Model Based Approach [Hua05, TAF12]	2005 - 2012	Maximum Likelihood, Multiple Imputation, Nearest Neighbor
Missing Values - Visualizations		
XGOBI [SCB98]	1998	Multiple Imputation, Scatter Plot
MANET [UHHS96]	1996	Multiple Imputation, Scatter Plot
Wang and Wang [WW07]	2007	Self Organizing Map, One Dimensional Graph (Histogram)
Templ et al.[TAF12]	2007	Aggregation Plot, Parallel Box Plots, Parallel Coordinates, Scatter Plots, Scatter Plot Matrices, Matrix Plot (Quality Band/Stripes)
Approaches by the partner	2013	Line Plots
Erroneous Values		
Sadiq et al. [Sad13]	2013	Statistical Error Detection Techniques, Constraint-BasedError Techniques
Outliers		
Achtert et al. [AKR$^+$10]	2010	Definitions
Sadiq et al. [HA04]	2004	Survey to Outlier Detection Methodologies
Charu C. Agarwal [Agg13]	2013	Book about Outlier Detection Methods
J. W. Turkey [Tuk77]	1977	Box Plot
Ieva et al. [IP15]	2014	Funnel Plot
Kim et al. [KK07]	2014	clustering algorithm and MST visualization
Liu et al. [LCW02]	2002	Self Organizing Maps
Novotny and Hauser [NH06]	2006	Output oriented method, data binning

2.4.4 Storage of Data Quality Information

During the state-of-the-art research only three resources were found that cover the question of how to store data quality information. For instance, the one of Xie et al. proposes to add data quality as a *new dimension*. Therfore, they add one data quality value for each data value in the original data set. This way, two vectors are created, one for the values of the record quality, as well as one for the dimension quality. The numerical values for those vectors are normalized and range from 0 to 1, where 0 is the lowest quality and 1 is perfect quality. However, this extension leads to an increase of dimensions (i.e., 2n+1) and by that of storage space needed. Figure 2.27 shows the resulting data structure of this approach.

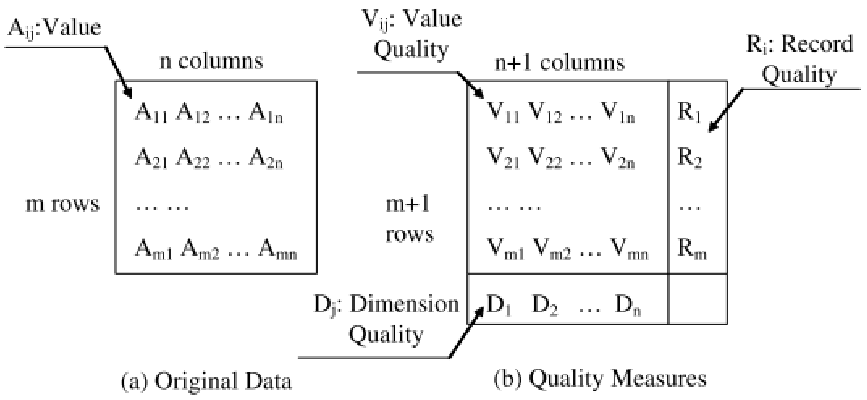


Figure 2.27: Xie's *„Data structure for storing data quality information"*[XHWR06].

Besides the efforts of Xie et al. and Rundensteiner et al. (see Section 2.3.2), which all make use of the concept of data quality as a *new dimension*, Huang [Hua05] raises the question of how to adequately store data quality information. In his contribution he argues that one might need at least one additional value to describe whether a single value is missing, accompanying the approaches of Xie et al. and Rundensteiner et al. Furthermore, Huang indicates that usually only a small part of the data values have data quality problems, which in turn means that the biggest part of the data values are of perfect quality (i.e., values of 1). However, in his work Huang ignores such scalability issues, since he only uses small datasets.

2.5 Summary / Conclusions

In Section 2.1, I first provided an explanation of how the resources reviewed were collected and categorized. I then gave a brief introduction to data quality by examining different definitions for data quality as well as for data quality measurement (see Section 2.2). From these I concluded that although common data quality dimensions exist, a common definition can not be easily given. This difficulty can partly be explained by the dependency of data quality on the respective problem as well as on the different user's needs involved. [God99, Sad13]. As a result, I investigated further contributions that are more specific to the introduced problem (i.e., for multivariate and time-oriented data) and data quality dimensions such as missing data, wrong data and implausible data [GGAM12] were discovered.

In Section 2.3 I diverged into multiple directions and reviewed contributions that indicate, encode or integrate data quality. Hence, in Section 2.3.1 I reviewed different approaches that „indicate data quality problems" on raw data, which led to the following conclusions. First, data quality visualization results in faster understanding of the actual data. And second, that data quality visualization generally improves the decision-making process. However, while all the contributions in this first category make it easier to identify data quality problems, they are not incorporated into the actual data visualization and by that make data quality assessment a separate task. More integrated approaches were then examined in Section 2.3.2, which all encode data quality into the actual information visualization. Amongst others, visual elements such as line width, blur, or hue were used in order to encode data quality in the actual information visualization. While not all of these approaches proved to be effective, most of them demonstrated that encoding data quality into the visualization further improves the decision-making process, as it lowers the cognitive demand and increases the decision accuracy [ZSC07]. One particular advantage of data quality encoded in the visualization is that it does not need additional space, and thus integrates seamlessly into the actual visualization. Conversely, other studies [SZ12] have shown that in cases of low complexity tasks, encoded data quality can also increase the cognitive load. The most advanced contributions were grouped in Section 2.3.3, which promote to integrate data quality information as a new dimension into the exploration process. This integration is often accomplished by adding linked interactive views that contain data quality as a new dimension. This new dimension is then used to interactively explore the actual data based on its quality. Compared to other approaches such as encoding data quality, incorporating data quality decreases the complexity of the decision-making task[ZWRS07, WXYR11] as well as it prevents cognitive overload (i.e., too much information in one place). Another important contribution in this category was the separation between the data space and the quality space [ZWRS07], as a fundamental prerequisite for interactive views.

One interesting finding from examining all these three categories is that only one single contribution addressed multivariate data [VdVK04] in combination with time-oriented data, while the majority solely dealt with multivariate data only. This can be also seen in Table 2.1, when comparing the columns *Mult.* and *Time.*. Moreover, most of the investigated contributions make use of parallel coordinates or scatter plot matrices in order to explore multivariate data. However, neither of these visualization techniques are an ideal approach for solving the problem introduced. Therefore, it is recommended to introduce adoptions such as additional views, which address time-oriented data and allow to traverse time. On the grounds of this identified lack of solutions for multivariate time-oriented data, I draw the conclusion that there is a research gap regarding data quality visualization in this field, as agreed by others [GGAM12, Joh04]. For instance, Griethe and Schumann [GS06] argue that despite advances made in the field of information visualization, there is a real need for research on how to convey information „as truly as possible". Their identified issues, even though primarily related to uncertainty visualization, to a large extend also reflect the current gaps in multivariate time-oriented data visualization. For instance, they named scalability issues with large data sets, asked how to present higher dimensional uncertainties as well as demanded to evaluate techniques that are the most suitable/reasonable?

After providing an overview of data quality visualizations for multivariate data I explored specific data quality visualizations techniques in Section 2.4 for the data quality problems identified in this work. First, I examined algorithms and visualization techniques that deal with missing values in Section 2.4.1. Therefore, I investigated deletion methods and soon noticed that they are no adequate solution in combination with data quality visualization, as they hide the absence of data from the analyst. Furthermore, amongst other algorithms I explored imputation and model based approaches, which are used to fill in the missing data gaps. However, since the actual algorithm is not in scope of this work I inferred that a convenient way to not hide the data's imperfection is to use an imputation algorithm such as dummy variables. Moreover, I found visualizations such as an *aggregation plot* (see Figure 2.17) and a *matrix plot* (see Figure 2.18), which in my opinion are promising ways to visualize missing values in overview[TAF12]. Additionally, I found approaches to visualize missing values in detail. For example, to use gaps where values are missing in a line chart, or to draw underlying bars instead of these gaps. For invalid values one contribution was found [Sad13] that addressed this quality problem. While the authors of this resource favored statistical error-detection algorithms over constraint-based error-detection algorithms, the latter was advocated. The reason for this is that constraint-based error-detection, deciding whether a value is an outlier by comparing it to a valid range, better matches the problem introduced. Consequently, no specific visualization technique for invalid values was found. in Section 2.4.3 I then examined algorithms and visualization techniques for outliers. The result of this is that besides a variety of algorithms to detect outliers (e.g., statistical models, linear models, etc.) I also found a few techniques for visualizing them. For instance, the one that I consider to be most interesting is the *funnel plot*, introduced by Ieva et al. [IP15].

To sum up, in each case the algorithms used exclusively depend on the problem as well as the data used [Hua05]. As a result, I have proposed both, the most convenient algorithms as well as the most desirable visualization techniques to be used for the proposed prototype of this work.

CHAPTER 3
Design

In this chapter I elaborate on the rationales for the technical architecture and the visual design of the proposed solution. For this, I first give a brief introduction into the technical architecture of the solution by explaining its main components. And second, I describe the individual views of the visualization, while I investigate their different design options as well as their implications.

3.1 Design of the Technical Architecture

The following three components form the basis of the technical architecture (see Figure 3.1):

- D3.js (JavaScript library for web-visualizations)
- Dropwizzard (Jetty, JAX-RS, Jackson)
- TimeBench (Java Library)

The first idea of how to tackle the business problem (see Section 1.1) was to use TimeBench, a Java library to create desktop information visualizations. This library was created by former members of the Information Engineering Group (IEG), which is a part of the institute of Software Technology & Interactive Systems (ISIS) [1] at Vienna University of Technology. At first, it was self-evident to use TimeBench to solve the business problem, but after reviewing several other technologies and comparing them with TimeBench it transpired that the opposite is true. The review showed that TimeBench's strengths lie in transforming raw data into temporal data, but not in visualizing this data on the

[1]more information about ISIS can be found here: `http://www.isis.tuwien.ac.at/` (retrieved June 05, 2014)

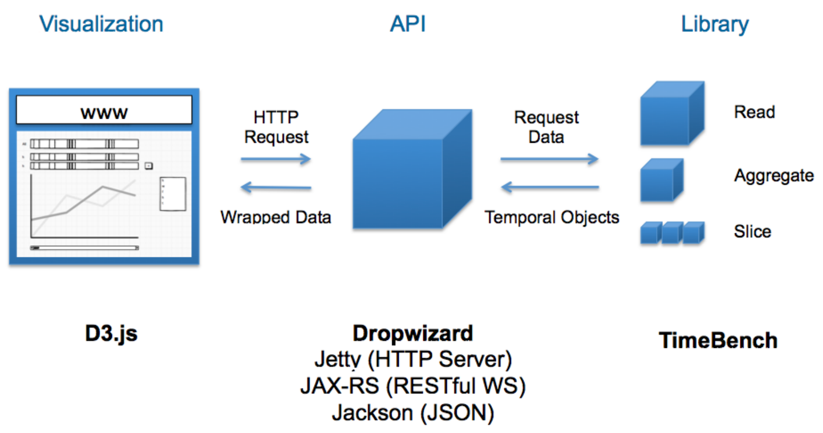


Figure 3.1: The technical architecture of the proposed solution.

web, which today becomes more and more important. Additionally, TimeBench's lack of documentation was another counter argument for using it to visualize the transformed data. At this point I had only one library that took care of data transformation, but I still needed to find one for the actual visualization.

During the comparison of different technical solutions to visualize data, I noticed one established library, namely *D3.js*. D3.js is a low-level JavaScript library for visualizing data with HTML, SVG, and CSS, primarily used to create web based information visualizations. However, when faced with large data sets, D3.js has its performance drawbacks when loading the data. For instance, to load the .csv file of our business problem takes more than ten seconds. This waiting time is considered very slow in terms of latency and perceived was a *disruption*, according to the authors of *Designing for Interaction*[Saf06, p. 150]. This latency problem led to the conclusion that pre-aggregating and reducing the data might result in a faster latency. A subsequent problem with this approach is that the client then has to load, aggregate and reduce the data each and every time it opens the application, which would also result in *disruptions*. Thus, I divided the different responsibilities (e.g., loading, aggregating, visualizing, etc.) among different parties (i.e., a client and a server) to overcome this problem.

3.1.1 The Server - Aggregating the Data

One way to archive a significant reduction of latency is to provide the client with data that has already been loaded, pre-aggregated and reduced in size. Those first steps require to either develop an own solution or to choose an existing library that provides those operations out-of-the-box. As the main focus of this thesis lies not on loading, aggregating, or reducing the data, a decision in favor of the latter option was made. The primary reason for choosing TimeBench was its capability to aggregate time-oriented data out of the box. Further arguments for TimeBench are the possibility to consult the authors directly, which makes it also easier to influence the development direction of the library. This influence makes it also more convenient to learn and use the library. A counter argument for TimeBench is the fact that it is based on Java, which is usually not the first choice when dealing with data transformations. However, I concluded that in our case the out of the box data aggregation capabilities are more important than the performance considerations.

3.1.2 The API - Accessing the Data

TimeBench was originally developed as an information visualization solution, which means that it does currently not offer convenient access to the data (e.g., with an API). This lack of access led to three different architectural design questions that had to be answered in order to expose the data to the client.

One of the most common ways to expose data on a server is to use a web-service. Hence, the first design question was *which web-server to use* in order to run a web-service on it. In our case, I decided to use *Jetty* as a solution for this. The main reason for this was that Jetty is based on Java and therefore integrates well with other Java based libraries such as TimeBench. Additionally, Jetty is very lightweight when compared to other solutions such as *Tomcat* or *Glassfish*, which makes it also easier to use.

The second design question I had to answer was *whether to use a RESTful or a SOAP based web-service.* Because D3.js can easily generate GET and POST requests, as well as dealing with JSON requests, I decided in favor of the former. Thus, I investigated different RESTful implementations and discovered *Jersey*, the JAX-RS (Java API for RESTful Services) reference implementation.

The last of the three questions was *how to automatically parse the client's requests and responses from the JSON format to native Java objects and vice-versa.* During reviewing I discovered *Jackson*, another Java-based library that automates this transformation, as a solution to this problem.

Finally, the deduced architectural questions were answered by the following three technologies:

- Which web server? → Jetty
- Which web service technology? → Jersey
- Which protocol to transfer data? → Jackson

Nevertheless, after answering those questions, it did not seem very practical to set up and connect all the libraries individually. Hence, I investigated whether there exists an all-encompassing solution that makes it easier to use all the three technologies. Fortunately, I discovered Dropwizard[2], a library that connects and pre-configures Jetty, Jersey and Jackson.

[2]The library is accessible under: `http://dropwizard.io/` (retrieved June 18, 2016)

3.1.3 The Client

So far, the technical solutions within the prior two subsections took care of loading, aggregating and exposing data. However, none of the mentioned technologies actually visualizes the data. This is where D3.js comes into play. During the last few years visualizations more and more moved from desktop solutions to the web. For instance, all the major visualization tools (e.g., Tableau, Spotfire, etc.) nowadays offer cloud or web based products, which gain popularity. This shift can be primarily attributed to technical advances of web technologies. An indicator for such a shift is the increasing number of solutions that are based on web technologies, presented on conferences and in papers. This increase in popularity of web based visualization frameworks and libraries lets us conclude that using *D3.js* to visualize the data is both more convenient and future-proof. The former argument is backed up by the fact that today every personal computer comes with a web-browser pre-installed, which makes the need for installing additional software obsolete.

3.2 Visual Design

In this section I demonstrate the various ideas and options that led to the final visual design of the proposed solution. Therefore, I provide several sketches that were created throughout the design-process in order to make it easier to follow. As a result of the state-of-the-art review (see Chapter 2), I first decided to separate the final solution into two separate parts (see Figure 3.2). First, into a part that provides the user with an overview of the data's quality. I call this part *quality view*. And second, into a part that provides the user with the actual visualization of the data itself, which I call *detail view*. Furthermore, from the review of the several solutions that visualize multivariate and time-oriented data, I concluded that both parts (i.e., views) need to be linked together. This linkage is provided by time, which means that traversing through time in one view also affects the other. One of the main reasons why I decided to have two separate views was the contribution of Xie et al. [ZWRS07], who divided their information visualization into two separate parts, namely the *data space* and the *quality space*. This choice was further advocated by other contributions that partly based their work on Xie et al. and also made use of separate views [VdVK04, RWX$^+$07, Hua05]. In this context, the proposed *detail view* corresponds to the former, whereas the *quality view* corresponds to the latter. Additionally, an argument for using a separate view for the data's quality was the contribution of Ward et al. [WXYR11] who advocate the use of a separate view, as it reduces the cognitive load. Ultimately, this separation is further supported by other contributions such as the design mantra of „Overview First, Zoom and Filter, Then Details-on-Demand“ by Ben Shneiderman [Shn96].

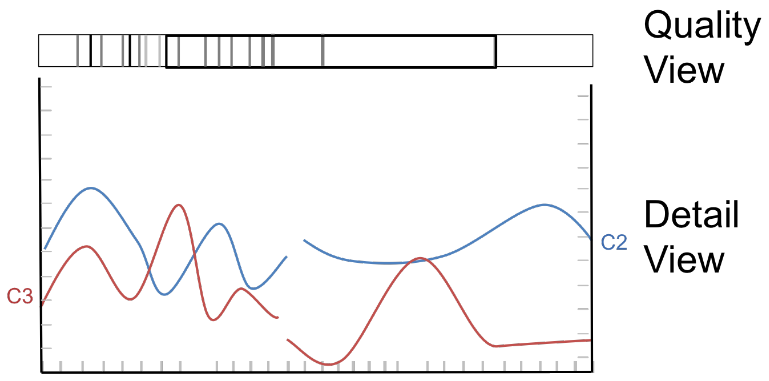


Figure 3.2: A sketched draft of the overall visualization.

3.2.1 Quality View: Visualizing Data Quality in Overview

Semantics

The *quality view* gives the user an aggregated view of the full range of the data's quality.

Initially, the quality view shows only one stripe with the label *„All“*. This stripe represents all the data-quality problems aggregated for all channels (see Figure 3.3) and consists of the following elements (see Figure 3.3):

- A *stripe* that consists of aggregated data-quality indicators for all channels
- A *toggle* that says **All**, which can be used to expand all the individual channels.
- A *slider* that indicates the current position in the linked details view (see Section 3.2.2)
- A *legend* which gives information about the different data quality levels
- A *dropdown* that lets one choose between different data quality indicators.
- A *tooltip* that is shown when the user points the cursor over a certain time slice.

The main ideas were drawn from Vlag et al. [VdVK04], which suggest to use a so called *temporal ordered space matrix*. This space matrix is an additional view, which displays the quality of the data for each year in overview (see Figure 2.8). Another inspiration for the quality view was the *matrix plot* introduced by Temple et al. [TAF12] which shows data quality values per variable, similar to the channels used in this work. Moreover, further design options to display data quality indicators in overview were discussed during the evaluating expert session (see Chapter 5).

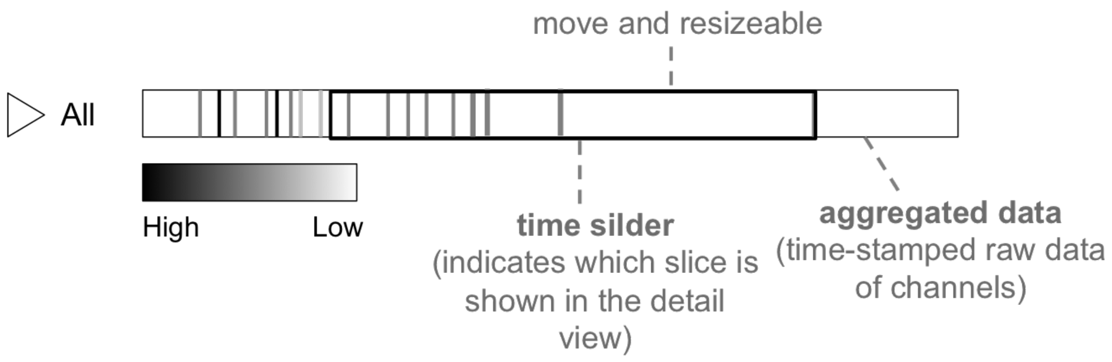

move and resizeable
All
High
Low
time silder
(indicates which slice is shown in the detail view)
aggregated data
(time-stamped raw data of channels)

Figure 3.3: A sketched overview, demonstrating the semantical elements of the quality view.

Interaction

The quality view offers the user methods to interact with its data (i.e., zoom in/out, filter, etc.) and the data displayed in other views, such as in the *detail view* (see Section 3.2.2). Thereby, the view follows the earlier mentioned mantra by Ben Shneiderman[Shn96]. In addition, the majority of the solutions identified throughout the state-of-the-art research support interaction [WXYR11, CWRY06, Hua05, VdVK04, RWX+07] as well as a few advocate the concept of linked views [KPP+12, TKSK08, Hua05].

Toggle: The „All“ toggle offers further ways of interaction. For example, by clicking on it the user expands the „All“ stripe and reveals the underlying channels (see Figure 3.4). In order to display only certain channels within the *All* stripe, the user can use the channel labels to either enable or disable channels individually. Another design option to in- or exclude channels would be to use check-boxes. For instance, these check-boxes could be placed right next to the respective label. Additionally, when all the channels are expanded, the user can use drag-and-drop to reposition the individual channels (see Figure 3.4). This repositioning might be especially helpful when one wants to conduct a side by side comparison in order to spot correlation.

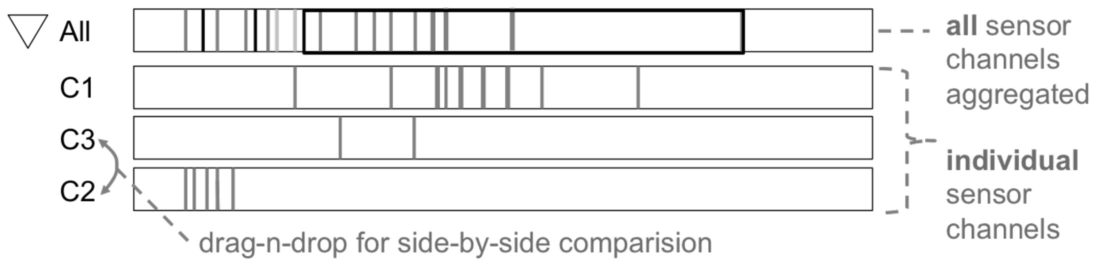

All
C1
C3
C2
all sensor channels aggregated
individual sensor channels
drag-n-drop for side-by-side comparision

Figure 3.4: A sketch outlining how the individual expanded channels might look like.

Slider: Another interaction method that a user would probably notice first is the *black border*, which contains areas of the stripe (see Figure 3.3). This *black border* can be used as a slider that directly manipulates the data displayed in the *detail view* and that can be used to traverse through time. The slider can also be dragged and resized along the x-axis according to the data slice of interest, which then automatically updates the data displayed in other views. Furthermore, it is also possible to pan and zoom, which accordingly updates the slider and the data slice shown. The idea for providing a slider was drawn from the contribution of Vlag et al. [VdVK04] as well as Huang [Hua05]. Another argument for using such a slider was that it provides a clear context (local detail) for the detail view, while the neighboring data sections can be guessed (global context).

Dropdown: Initially the stripe in the quality view only shows all the data quality indicators in an aggregated way. Thus, if the user wants to only display a specific data quality indicator he or she can use the provided dropdown menu. This dropdown lets the user choose between the different data quality problems, as mentioned in Section 1.1.2. The idea for the dropdown was drawn from the contribution of Führing and Naumann [FN07] who offer both, the rating of individual criterion, as well as to rate and display a global criteria, which covers (i.e., aggregates) all individual criteria. One design option is to color-code the different data quality indicators and use different shades to indicate the frequency of their occurrences (see Figure 3.5). Another option is to change between data quality indicators using gestures, where the user swipes over the stripes and by that changes between different indicators. However, the solution is a desktop visualization, why I conclude that the design option can also be considered a matter of future research.

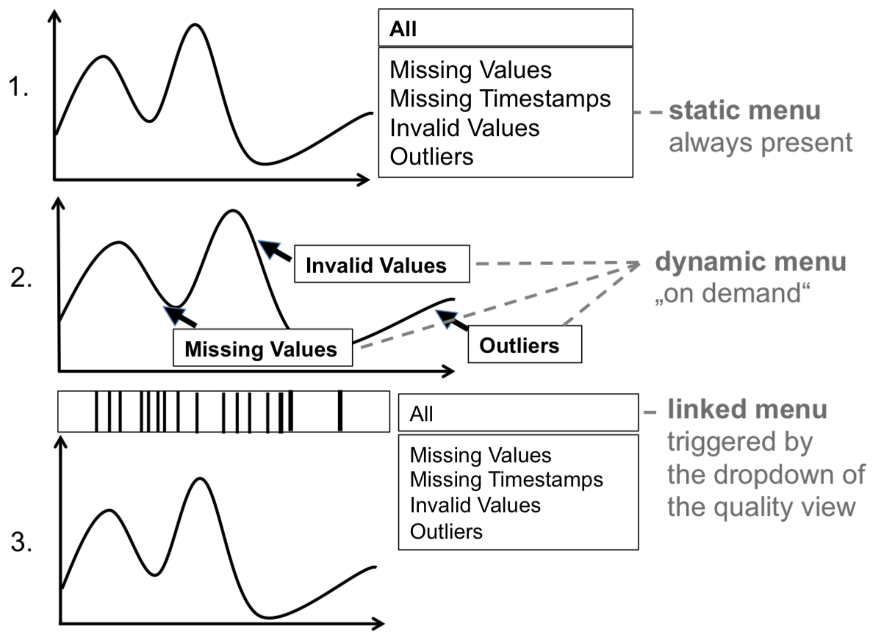


Figure 3.5: A dropdown to switch between the different data-quality indicators.

Tooltip: In order to also cover the third aspect of the mantra by Ben Shneiderman, a *tooltip* that offers *„Details on Demand“* was introduced. By positioning the cursor over a certain area of the data quality stripe, the user obtains the following detailed information:

- start and end time of the respective time slice (i.e., of the area)
- length of the time slice
- channels and data quality indicators that caused the problems

The first two items are displayed in all the different tooltips (see Figure 3.6), whereas the third depends on the stripe over which the cursor was positioned. This means that it depends on the combination of channel and data-quality indicator that was selected. The following combinations with the additional information are possible (see Figure 3.7):

1. **All Channels/All Indicators:** additionally shows the channels which contain data-quality problems, with labels that are color-coded according to the data-quality indicators
2. **All Channels/One Indicator:** additionally shows the channels which contain data-quality problems, but without color-coded labels
3. **One Channel/All Indicators:** shows only the data quality indicators.
4. **One Channel/One Indicator:** no additional information, as the channel and indicator are determined.

In the case of the first two combinations only the most contributing channels are displayed. If the number of the causing channels exceed the number of three the tooltip displays *„others...“* right next to them to indicate that there is more than three channels with data-quality problems.

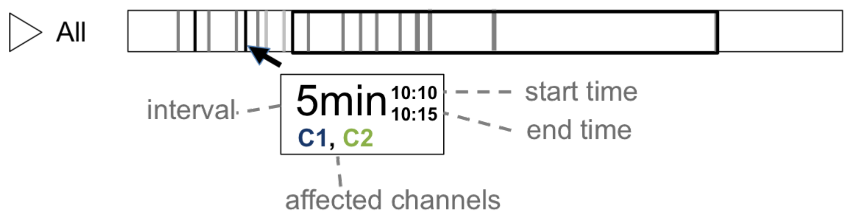


Figure 3.6: A sketched draft of tooltips that provide the user with details on demand.

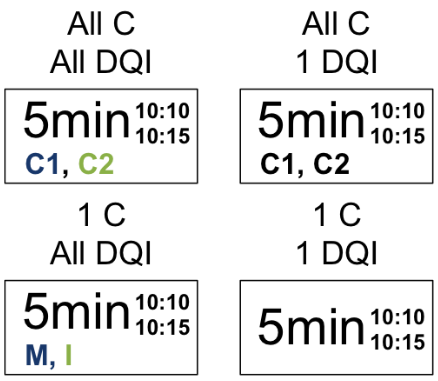


Figure 3.7: All tooltip variations, „*C*“ = Channels and „*DQI*“ = Data Quality Indicators.

3.2.2 Detail View: Visualizing Data Quality in Detail

The main view of the proposed solution is the *detail view*, for which it was decided to use a line-chart that combines up to two superimposed channels (see Figure 3.8). The decision for a line chart was made, because it is an intuitive way to traverse time and it works for most of time series data, whenever one deals with a lot of data points [Yau10]. Moreover, a central argument for a 2D visualization, such as a line chart, was provided by the contribution of Huang et al. [Hua05]. In their contribution they support the use of the two dimensional space in combination with the visualization of data quality, as they evaluated the third dimension to be a poor choice. On the other hand, line charts also have their limitations. For instance, when displaying several channels at the same time, they make it harder to explore multivariate data. Hence, I added an additional interactive view (i.e., the *quality view*) in order to partly address this disadvantage. However, the main purpose of the *detail view* is to compare occurrences of data quality problems within and across channels (i.e., to spot correlations). Furthermore, the *detail view* is synchronized with the *quality view* (see Section 3.2.1), which means that zooming in the *quality view* automatically updates the data in the *detail view*.

Semantics

The detail view consists of the following parts:

- one horizontal axis for the time (x-axis: time)
- one or two vertical axes for the displayed channels (y-axis: values)
- scaled ticks, representing time units
- data quality indicators
- direct labels (incorporated in the graph) for the channels

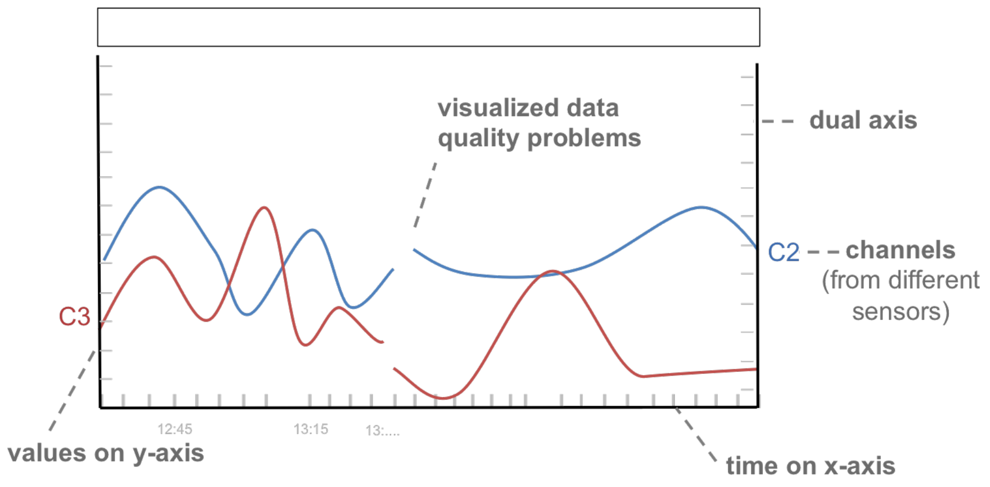

visualized data quality problems
dual axis
C2 – – channels (from different sensors)
C3
values on y-axis
12:45
13:15
13:....
time on x-axis

Figure 3.8: A sketched overview of the semantical elements of the detail view.

- eventually a separate quality view, showing the data quality issues on a granular level.

The initial idea of the detail view was to incorporate up to 5 channels, but this would have led to a decrease in simplicity and usability, as it would ultimately add clutter to the visualization. One way to overcome such clutter is to make use of the principles behind an approach called small multiples. The principle in its essence means to place multiple line charts one below the other. I advocate that having multiple charts with only a few channels is the cleaner and more flexible solution [Val14, Gem10]. Moreover, I decided to allow only two different channels per chart, but extend this by allowing multiple charts in order to still be able to compare more than just the two channels. One substantial argument for only allowing two channels is the fact that it becomes harder to also display their respective axis, whereas with only two channels this is straightforward. In the following paragraphs I elaborate on the design options regarding displaying the different quality indicators (i.e., missing, invalid, outliers) in the detail view.

Missing Values and Timestamps, as one of the three major data quality problems presented in this work, are problems where I found only a few solutions during our state-of-the-art research. The majority of the resources found (see Chapter 2) use imputation to estimate and substitute data defects, instead of visualizing them. Hence, I considered the following ways to indicate missing values (see Figure 3.9):

- use a bar to indicate a missing value
- leave the value out and display a gap, provided that the level of detail allows it, otherwise indicate a missing value by setting a marker at the x-axis.
- substitute missing values with imputed values and color code for its absence.

- a combination of the above

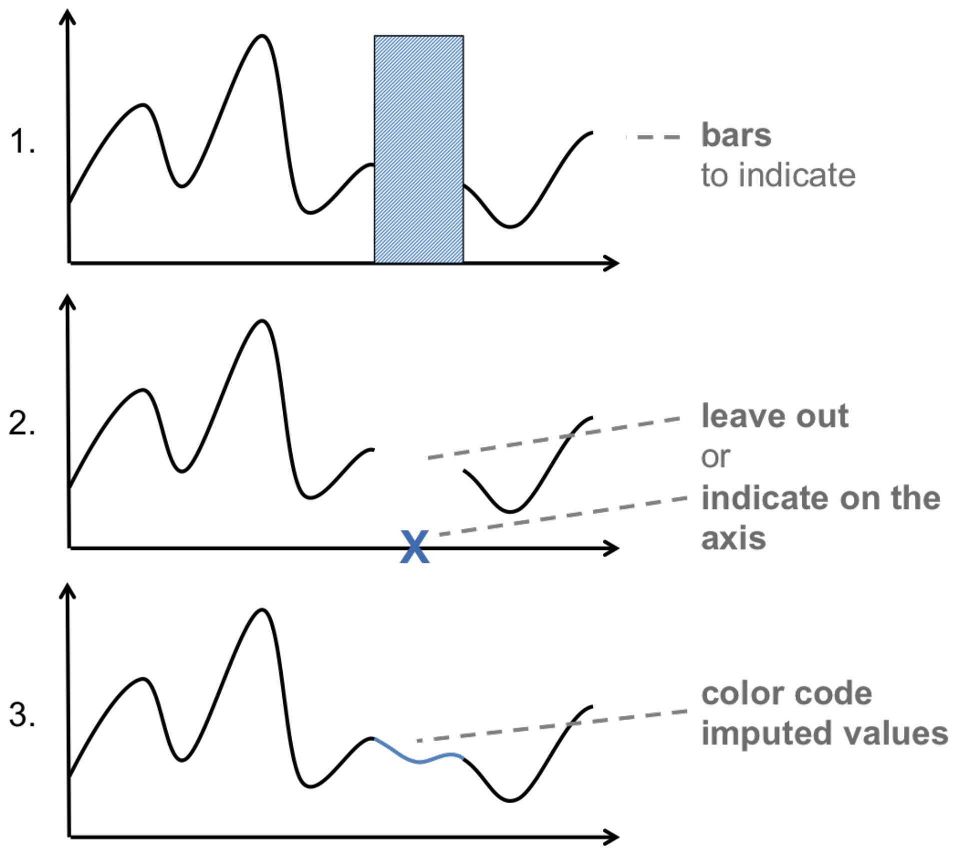


Figure 3.9: A sketch demonstrating the different options to display missing values in the detail view.

Nevertheless, all these described methods to indicate missing values do not solve the problem of missing timestamps, which is the absence of a whole data row (i.e., measurement). This absence would only be visible if one displays all channels at the same time. For instance, if I visualize missing values by using the second design option, then the user would notice a missing timestamp because of a complete vertical line that would indicate no data, given the right zoom level. If the chart does not display data at the right zoom level then one could still use markers, which indicate missing timestamps.

In our case **Invalid values** are values that exceed a certain threshold (see Section 2.4.1). For instance, it usually is known whether a certain channel (e.g., the pump pressure) can transmit values below certain values (e.g, zero bar). Thus, if such a channel transmits a value of -1, it can be assumed that the respective measurement is invalid. In the following I describe different options that I considered in order to indicate and visualize invalid values in the *detail view*.

1. leave the value as it is and use color coding.
2. leave the value out, but indicate its invalidity.
3. substitute invalid values with imputed values and color code for its invalidity.
4. mark the invalidity on the axes.

The first option has the significant disadvantage that high or low invalid values completely distort the scale of the data and, by that, question judgments made on the basis of such data. Moreover, I argue that such invalid values do not give additional information of the current state, which is why maintaining the scale of the data might be of more importance. The second option connects the values of the predecessor and successor of the invalid measurement and color codes the resulting part of the graph. In contrast to option one, option two maintains the scale of the data, but does not give information about where the data should be. The third option is similar to the second, but uses imputation to substitute invalid values with estimated. I therefore decided to use option three, as to me it seemed the least distracting.

Outliers are another particular problem when dealing with data quality issues. In the past, outliers were often refereed to as very high or low values. This misleading description of outliers contributed to early approaches which dealt with outliers by simply cutting them off. On the other hand, our proposed solution aims to keep the overall image of the data as unbiased and undistorted as possible. Thus, instead of cutting values off or manipulating the raw data, I determined the following ways to visualize outliers (see Figure3.10):

1. using areas that show the normal range of the data
2. color coding of values and parts of the data that are potential outliers
3. instead of using areas one could also use so-called outlier bands, which also take the development of the measurements into account.

The first option is rather unpretentious, but does not take „locality“ of data values into account. This means that it assumes everything to be an outlier which is below or above a certain threshold. However, in the state-of-the-art research (see Section 2.5) I found approaches which do and by that prove such simple assumption wrong. The second option, which is to use color coding, appears to be straightforward, but in our opinion distorts the big picture. Moreover, to solely use color coding might result in an overuse of this method, if applied to every data quality problem. In contrast to the first two option, the third takes locality into account and by that is the most exact technique to visualize outliers. However, this advantage comes at the cost of adding the most clutter to the chart if compared to the other two options.

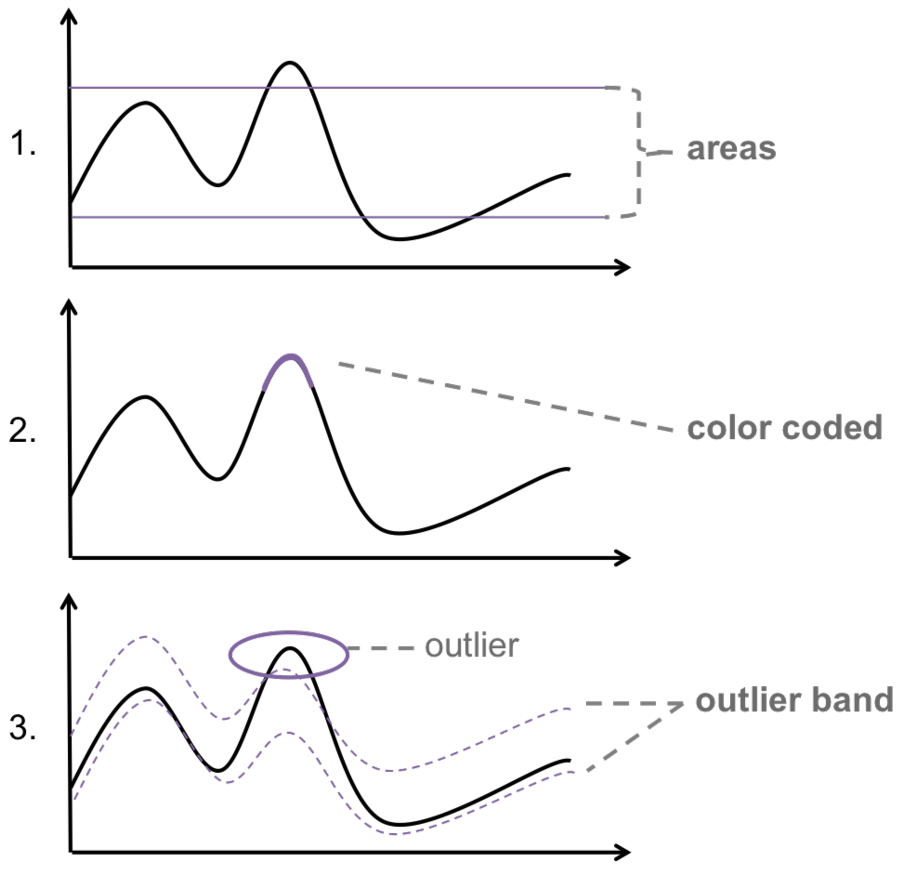


Figure 3.10: A sketched overview of the different options to display outliers in the detail view.

Dropdown: After discussing the semantics of the different data quality indicators in the detail-view, the questions of how to switch between those indicators arises. To switch indicators the following options were considered:

1. A static dropdown menu that is placed right next to the chart. This option has the advantage that one can easily change the data quality indicators for both of the displayed channels, as well as for all the different detail views separately. One disadvantage is that there is no simple way to change the data quality indicators for one channel only.

2. A dropdown menu on demand that appears whenever a user clicks on one of the channels (see Figure 3.11). This has the major advantage that one can select the data-quality indicators for each channel individually. Additionally, the design of this option is cleaner compared to a static dropdown. A disadvantage is that one can then not change the quality indicators for all the displayed channels at the

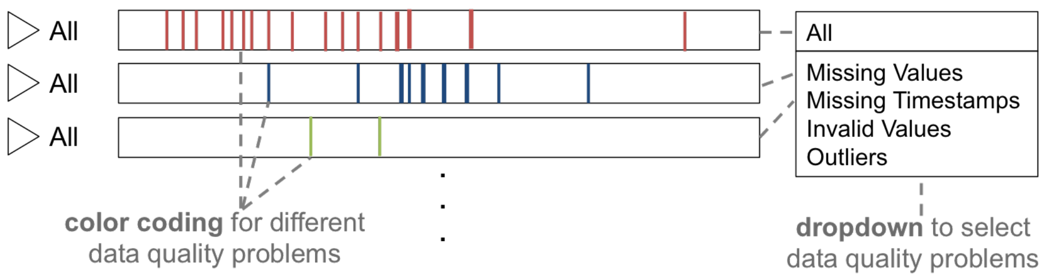

All
All
All
All
Missing Values
Missing Timestamps
Invalid Values
Outliers
color coding for different data quality problems
dropdown to select data quality problems

Figure 3.11: A sketch that shows the design options for selecting a data quality indicator for the detail view.

same time, because a single click is only associated with one channel only. Thus, changing the data quality indicators for multiple channels becomes inconvenient.

3. Linkage of the dropdown menu of the quality view with the ones of the detail views.

After reviewing the different design options in the face of the initial user tasks, I conclude that the fourth option (i.e., Linkage) is the most suitable. Even though this option offers the least flexibility in terms of comparing the different channels, it is the least obtrusive. Moreover, having only one dropdown for selecting the data quality indicator makes the interface simpler and cleaner. Advanced elements, such as the first two options are therefore matter of future research.

Quality Views for each Detail View? During discussions regarding the detail view in a group of information visualization experts the following question came up: *„Should each detail view has its own quality view?“*. One essential argument for having a separate quality view is the ability to then see data quality indicators also on a more granular level. Another aspect of having a separate quality view is that the data quality markers become unnecessary, as the same information is then displayed in the respective quality view. Moreover, from the initial task we inferred that having an additional quality view for each chart only has a benefit when the user only deals with one single channel. This means that as long as only one single channel is displayed in a chart an additional quality bar will be added. As soon as the user adds another channel to the same chart the first quality bar will be removed. In case this additional quality bar is perceived as a distraction, the user is provided with means to hide it. For instance, if the user fully zooms in on the data then the whole quality view in detail will be displayed as a single bar with only one color, which has no benefit for the user.

In the course of answering this question, further questions arose. For instance, „when having separate quality views should one also have separate sliders to traverse times in the detail views?“ A separate slider would give the user the ability to compare channel pairs at different points in time, otherwise one could only compare channel pairs at one point in time. Thereby, adding a separate slider has the implications that it would increase the flexibility to compare channels, the time that is needed to change all the different sliders, as well as it would add more clutter to the visualization as a whole. Nevertheless, as the initial tasks do not require such a flexibility with comparing channels over different times I draw the conclusion that separate sliders are not necessary, as they would only add clutter to the visualization.

Interaction

Add/Remove Channels: the first interaction method a user will notice is to add channels. At the beginning, the detail view only shows a sample graph and a big plus button on top of it. The user can then add new channels either by dragging and dropping them from the data quality view, or by using the plus button. In case of the latter the user will be provided with a complete list of all available channels that can be added to the detail view (see Figure 3.12). To remove a channel one has to click on the channel which will reveal an x-symbol right next to the graph. By clicking the x-symbol one removes the graph from the detail view (see Figure 3.13).

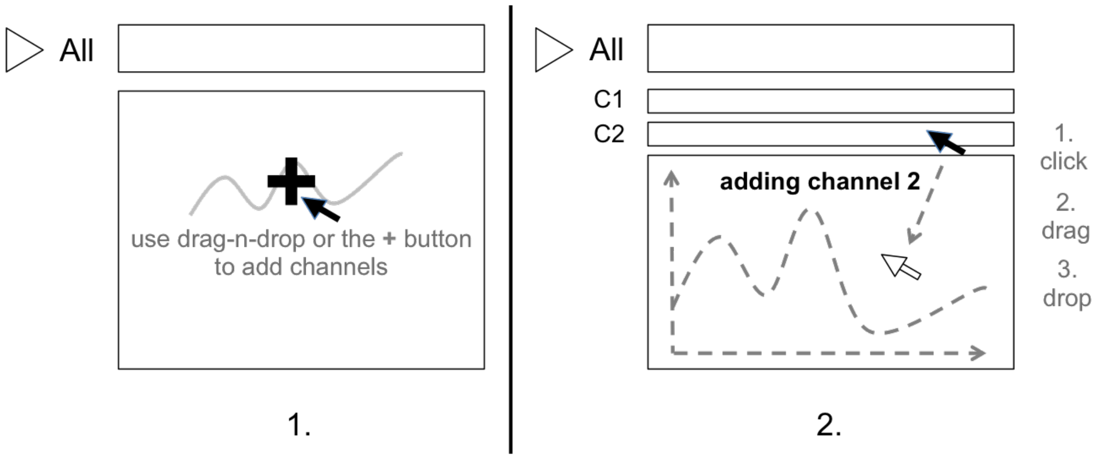


Figure 3.12: A sketch that demonstrates the different options to add a channel to the detail view.

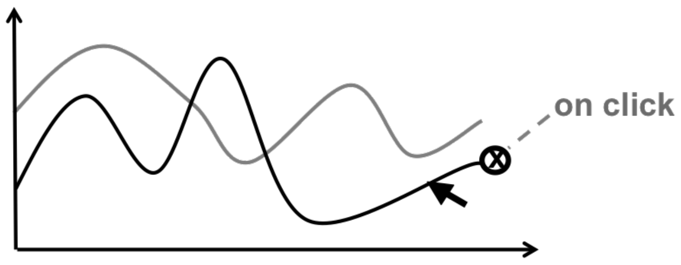


Figure 3.13: A sketch that shows how one can remove a channel from the detail view.

Zoom in/out: following the design mantra, another step in exploring the data is to zoom in/out. For this interaction the detail view offers several methods. For instance, one can use the slider provided in the quality view to either zoom in or out of the data as well as to traverse through time, by moving it to the right or to the left.

Switching between Quality Indicators: can be done by using the dropdown menu in the quality view, which is linked to the detail view. Initially, „*All*“ is selected in the quality view, which leaves two different design options. The first option would be to display no data quality problems until the user selects a specific data quality problem. The second option would be to display all data quality indicators at the same time. One argument against option two might the fact that the interface then could be overloaded. However, since the domain expert can switch to individual data quality indicators, it was decided to leave all as a default.

Details on Demand: Additionally, the user can display details about the respective graph by either hovering over or clicking on the graph. The former method involves a box that appears next to the current cursor position and displays details such as the time slice of the data point, the data quality, or data quality issues if present. The latter method on the other hand displays detailed information about the respective channel rather than a specific point in time. This means that by clicking on a channel the user will be provided with the channel name, the data quality of the current time slice, as well as the start and end point of this time slice.

3.3 Summary

In this chapter all the individual parts of the proposed solution were introduced, as well as their underlying rationales. The final solution incorporates the previously introduced quality view, the detail view as well as their interaction elements. Figure 3.14 provides a big picture of all these proposed elements.

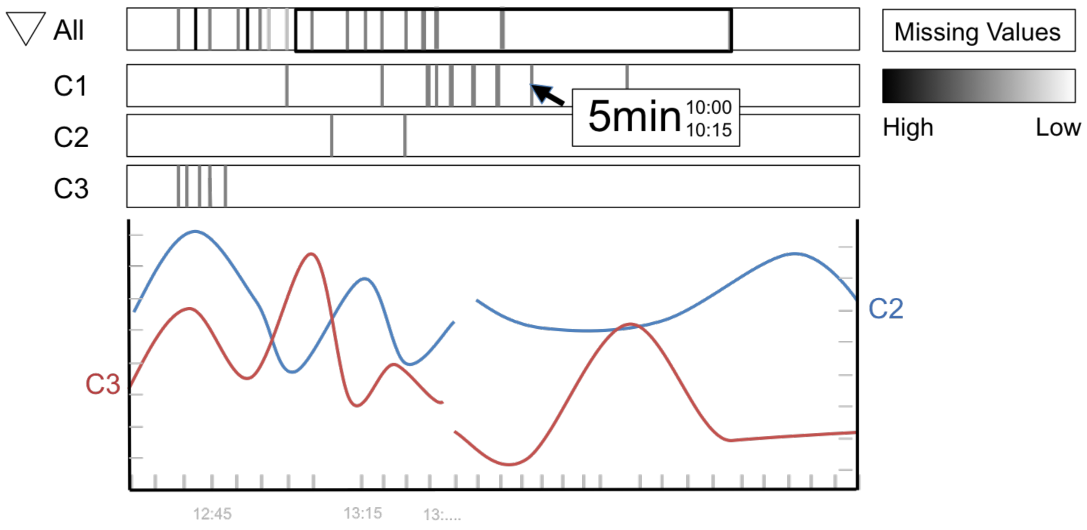


Figure 3.14: A big picture of the proposed design.

CHAPTER 4

Implementation

In the following chapter I elaborate on the challenges faced during the implementation of the prototype. These challenges can be placed into two separate groups. The first group consists of challenges related to the *technical implementation*, during which I had to deal with the question of „How to set up and combine the libraries chosen in the design chapter“ (see Chapter 3). The corresponding Section 4.1 covers aspects from loading the raw data to transforming and transmitting the data to the client. Part *c* of Figure 4.1 represents the most relevant parts of the loading process, while part *b* shows the most relevant parts that address the transformation and transmission of the data. The second group, on the contrary, focuses on how d3.js can be used to visually address the data quality problems introduced (see Section 1.1). Section 4.2 covers these aspects from visualizing the requested data and its data quality to handling the user's interaction between different views. For this second group, the structure is based on the structure of the design chapter in order to make a comparison of the proposed design with the implemented prototype easier. Part *a* of Figure 4.1 shows the most relevant parts for this second group.

4.1 Implementation of the Technical Architecture

4.1.1 Building the Foundation

The first step of implementing the prototype was to set up a new Dropwizard project that built the foundation of the proposed solution. This was necessary because Dropwizard built the link between loading and aggregating the data on the one side and visualizing the data along with its quality on the other, shown in Figure 4.1. As a consequence, Dropwizard acts as an entry point, as it invokes TimeBench code and provides the web service from which data can be requested. For this set up I first created a new Maven

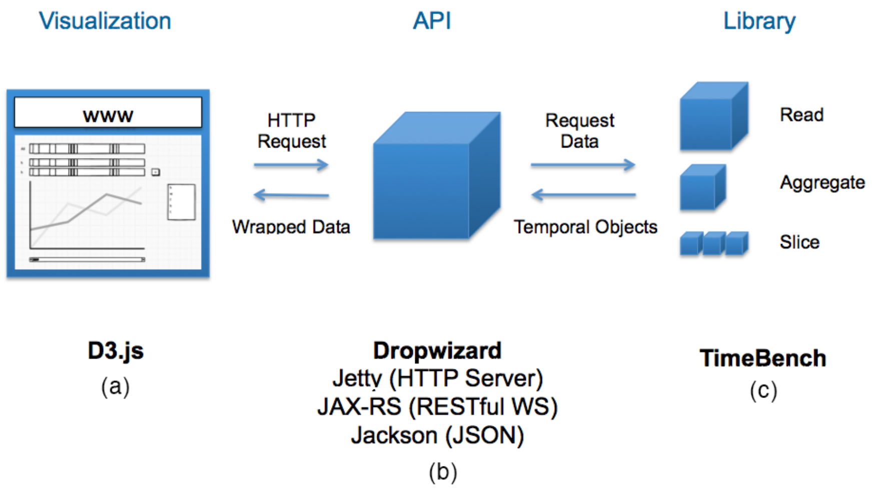


Figure 4.1: The architecture of the prototype as well as its individual parts a, b and c.

project and added both, the Dropwizard library[1] and the TimeBench library[2] by using their available Maven dependencies. In order to set up the necessary components of the newly created Dropwizard project I used the *getting started*[3] section on the web site of the library. Figure 4.2 shows the file tree of the prototype after the initial set up. Regarding this file tree, in the following I elaborate on the most important classes and packages that I used in order to set up the foundation of the prototype, that is the web server and the web services it provides.

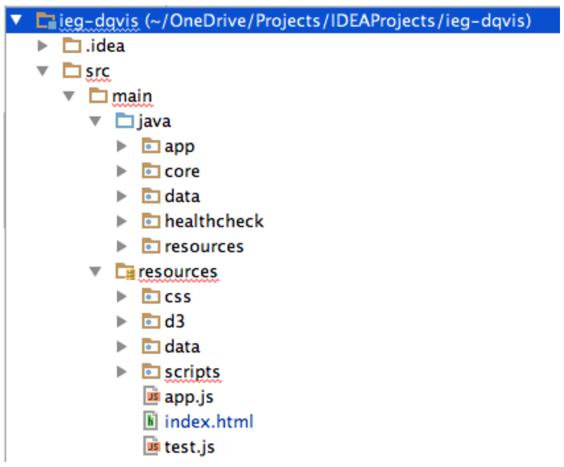
- **Configuration Class:** configures the application. This class is passed as a parameter in the *run* method of the *Application* class and therefore necessary to start the prototype, also shown in Figure 4.3. For this I created the *ApplicationConfiguration.java* class based on the *getting started* template.

- **Application Class:** starts and runs the web server and the web services. The *run* method of this class starts the JETTY web server and registers the JAX-RS web services. Figure 4.3 shows the dependencies of this class. This class has two important methods. First, the *initialize* method, in which a new instance of a DAO is created that reads the underlying data by using TimeBench. After such a new instance is created the data is loaded and then aggregated (more on accessing the data in Section 4.1.2). Second, the *run* method that creates and registers the necessary *Resource* classes (i.e., the provided web services).

[1] Maven dependency for Dropwizard: `http://mvnrepository.com/artifact/io.dropwizard/dropwizard-core` (retrieved July 10, 2015)

[2] Maven dependency for TimeBench: `http://mvnrepository.com/artifact/org.timebench/timebench` (retrieved March 21, 2015)

[3] Getting started section for Dropwizard: `http://www.dropwizard.io/0.9.2/docs/getting-started.html` (retrieved August 04, 2015)



```
▼ ieg-dqvis (~/OneDrive/Projects/IDEAProjects/ieg-dqvis)
  ▶ .idea
  ▼ src
    ▼ main
      ▼ java
        ▶ app
        ▶ core
        ▶ data
        ▶ healthcheck
        ▶ resources
      ▼ resources
        ▶ css
        ▶ d3
        ▶ data
        ▶ scripts
          app.js
          index.html
          test.js
```

Figure 4.2: Technical architecture of the prototype.

- **Representation Classes:** determine how the response to the clients looks like. Those classes, by using Java annotations and the Jackson library, define how the requested data is serialized between the server and the client. For the prototype I used four such representation classes, namely *TemporalData.java*, *TemporalColumn.java*, *TemporalValue.java* and *AffectedChannel.java*, which where furthermore nested. This way, a *TemporalData* element consisted of multiple *TemporalColumn* objects, which again consisted of both, a *TemporalValue* and a *AffectedChannel* object. These nested classes as well as their dependencies are depicted in Figure 4.3. The resulting nested data structure is constructed and populated by the aggregated data from TimeBench and invoked in the *Resouce* class in the Dropwizard project. The actual serialization of this nested data structure is automatically handled by *Jackson*, a library built into Dropwizard.

- **Resource Classes:** resource classes define which resources (e.g.; aggregated data slices) can be requested from the web service (i.e., the visualization layer). For this reason resource classes among other things contain annotated *GET* methods that determine which and how requests can be made as well as how the returned values look like. For the prototype I created a *TemporalDataResource.java* class that handles all requests and passes the parameters to the respective DAO methods. These methods then make use of TimeBench to access the aggregated data and its quality problems (more on accessing the data in Section 4.1.2).

- **Health Check Class:** give a way of adding tests to the application that verify that the application is working correctly.

In addition to Dropwizard and TimeBench, I also used the following frameworks/libraries:

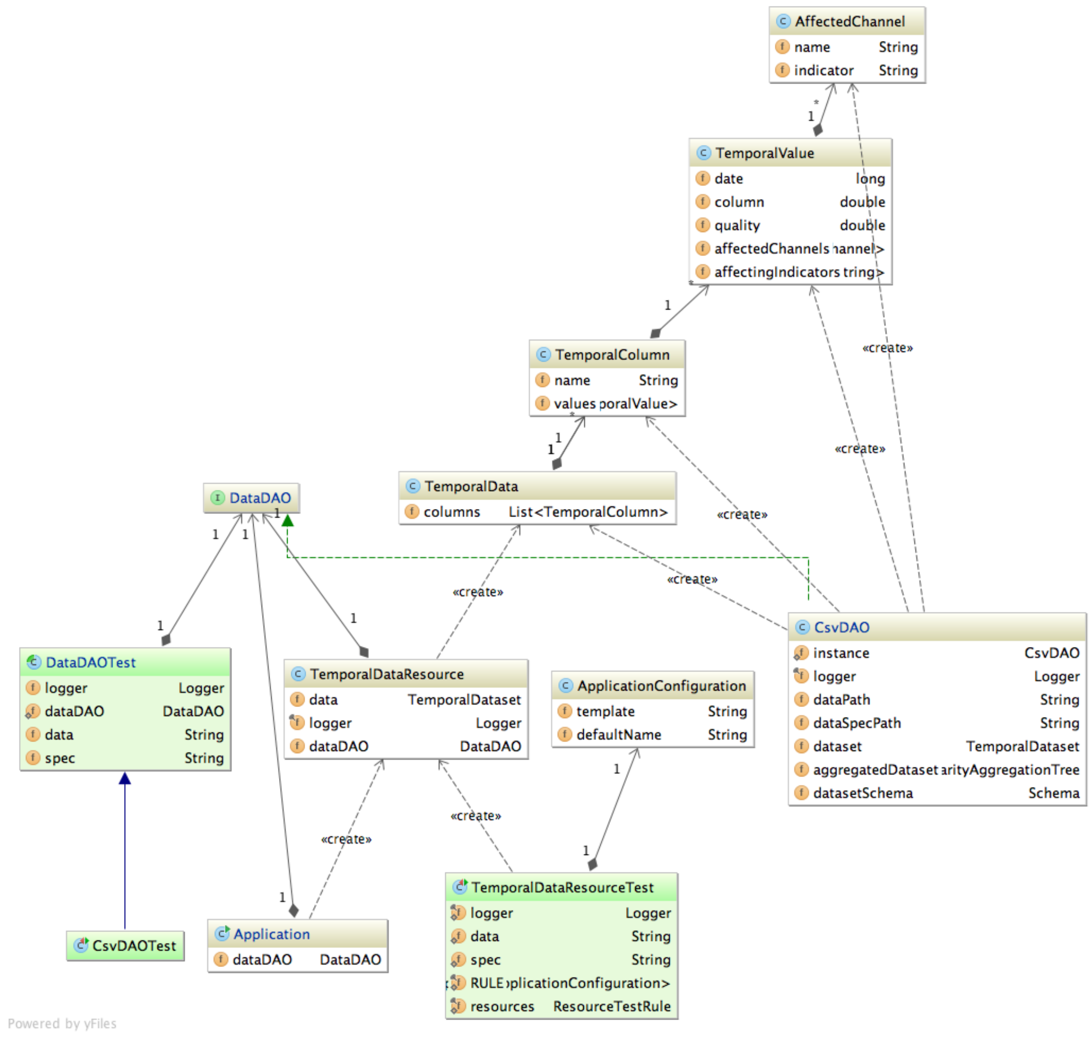


Figure 4.3: Most important classes of the prototype as well as their dependencies.

- **Maven**[4]: was used for Java Dependency Management.
- **Karma**[5]: for testing the JavaScript code.
- **RequireJS**[6]: used for JavaScript Dependency Management.
- **JUnit**[7]: was primarily used to Blackbox Testing.

[4]more details on Maven can be found at: `https://maven.apache.org/` (retrieved July 10, 2016)
[5]more details on Karma can be found at: `https://karma-runner.github.io/` (retrieved July 11, 2016)
[6]more details on Require can be found at: `http://requirejs.org/` (retrieved July 15, 2016)
[7]more details on JUnit can be found at: `http://junit.org/junit4/` (retrieved July 15, 2016)

4.1.2 Accessing the Data

In order to separate the higher level business logic from the lower level data accessing operations I used the Data Access Object (DAO)[Pan02] design pattern. For instance, if the underlying data resides in a database instead of a comma separated value file, it would only be necessary to implement a new DAO implementation class for databases instead of making changes throughout multiple classes. Therefore, I created a *DataDAO* interface, which determines the methods of such an implementation class, and a *CsvDAO.java* implementation class for the provided comma-separated data. This implementation class has the following purpose:

- *loading* and *storing* the raw underlying data of a csv file, given a file path and a TimeBench file specification.
- *aggregating* the raw data for all channels (i.e., columns), as well as for their data quality measures, given predefined granularities
- *reading* time slices of the raw as well as the aggregated data of all or selected channels, given a range.

In the following the most relevant code snippets for these three tasks are briefly explained.

Loading and Storing the Data

The initial entry point for loading the data is the *Application* class and its *initialize* method. As soon as the initialize method of this Dropwizard class is called, a new instance of the *CsvDAO* class is created. During the creation of this new *CsvDAO* object (i.e., in the constructor) the *readData* method is invoked, which utilized TimeBench's *TextTableTemporalDatasetReader* and ultimately loads the data. In order to create such a *CsvDAO* object two input parameters are needed. First, the path to the file containing the raw data and second, the path to a specification file. The latter describes the raw data. For example, the specification explicitly defines the temporal column. After the raw data is loaded it resides within a *TemporalDataset* object, which can be described as follows:

> The central class is TemporalDataset [...] Internally the temporal dataset is composed of a number of data columns holding non-temporal attributes [...] and a `TemporalColumn`. This specialized data column holds one reference to a temporal element for each row, which realizes the timing function `t`. The TemporalElement tuples, `E` , are created and stored by the `TemporalElementStore`... [RLA$^+$13]

At this point the data was loaded into the *TemporalDataset*, yet I still needed to aggregate it in order to extract and transform it later, which will be explained in the following.

Reading the Data

To this point the data is successfully loaded and stored in a *TemporalDataset*. However, one challenge that wasn't addressed so far was the volume of the data. For instance, if the raw data were loaded and directly used in the browser then *d3.js* would need to deal with approximately between 241.960 and 524.120 data points per channel, which is hard to visualize all at once as well as hard to grasp by a user. Therefore, an approach had to be found in order to compress the data transmitted, whereby the following was considered:

- **Loading Raw Data:** the first approach considered was to load the raw data into TimeBench and then directly pass it to d3.js. The built in functions of d3.js could then be used to overcome the visualization problem of high volume data. However, this did not solve the problem of transferring high volume data from the server to the client, which in our case can easily be up to 200 megabytes. As a consequence, such transfers inevitably lead to longer loading times and become unacceptable in terms of user experience. This was already confirmed in earlier experiences made with d3.js, where projects with data volumes of only a few megabytes resulted in loading times of more than 5 seconds. This was further confirmed by a set of articles discovered on portals such as *Quora* and *Stackexchange* [Zha14, Cho13, Div15]. As a result, the remaining question was how to reduce the size of the data before transmitting it from the server to client.
- **Loading Aggregated Data:** the second approach is to aggregate the data, and thus reduce the size of the transmitted data. In the case of the implemented prototype the temporal column was used in order to aggregate the data by the following chosen granularities:
  - **Top:** the raw data, with a time frame of 6 weeks up to 3 months this results in 241.960 - 524.120 tuples.
  - **Day:** on a daily basis this results in 42 - 91 tuples.
  - **Hour:** on an hourly basis this results in 1.008 - 2.184 tuples.
  - **Minute:** on an every minute basis this results in 60.480 - 18.719 tuples.

  While this aggregation already addressed the long loading times, data would still grow with a bigger data set, which leads back to the initial problem.
- **Loading Sliced Data:** as an enhancement to aggregation, a way to constraint the amount of data transmitted had to be found. For instance, the client would request and receive only data points of a single week instead of the whole 3 months. This way each of the requested data „slices" would not exceed defined size limits and the transmitted data could be displayed on the client's browser within an acceptable period of time. Even though time slices solve the problem of long loading times, they do not automatically allow for a *zoom in or out* interaction, as proposed in Section 3.2.2.

Finally, in order provide the user with *zoom in and out* interaction I combined the advantageous of *Aggregation* and *Sliced Data*. As a consequence, I made the aggregation level dependent on the range of the requested data slice. For instance, if only a small slice of the data is requested (e.g., one single day) then it is possible to transmit the raw data instead of the aggregated data. As a result, I concluded that the solution is to make the transmitted data dependent on both, the given range and the aggregation level. This means that if the whole data set is requested, which for instance contains data points for 3 months, then a bigger granularity such as days or even weeks will be ideal. In contrast, if the data for only one day is loaded, then a smaller granularity such as hours or minutes will be sufficient. This way the data to be transmitted stays within defined size limits and the loading times for the user are acceptable. Figure 4.4 shows an example, including the data points per channel for all the different aggregation as well as for a single data slice.

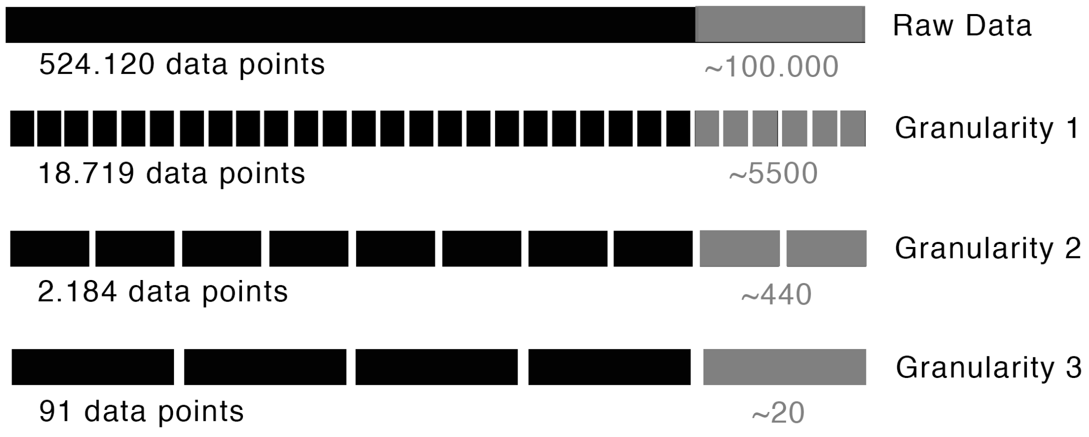


Figure 4.4: Number of data points resulting from the use of different aggregation levels.

Aggregate the Data

After the *TemporalDataset* is stored the *aggregate* method is called. This method utilizes TimeBench's *GranularityAggregationTree* data structure in order to aggregate the data. This data structure is created by using two parameters, a *TemporalDataset* object and a *Granularity* array. The former is created whenever a CsvDAO instance is created, whereas the latter is hard coded in the method. For the prototype I used minute, hour and day as such hard coded granularities. This GranularityAggregationTree then holds the aggregated data as well as the aggregated data quality measures. How this data is then read and transformed will be explained in the following.

Read the Aggregated Data

For reading the data I made use of TimeBench's *getNodeTable* method, which can be used on a DataAggregationTree. This method returns an Iterator which provides means to iterate over all temporal objects, given a certain granularity. Therefore, I included this iterator in my own *readAggregated* method in which I assembled the new data structure suitable for Dropwizard *Resource* to be accessed by a web service later on. The main challenge in combination with constructing this data structure was to calculate the data's quality *dq*. The general formula for the calculation is defined as:

$$dq = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij}}{n \cdot m} \tag{4.1}$$

where n is the number of selected channels and m the number of included data quality problems.

An example for calculating the data's quality (see TemporalElement 4 in Table 4.1):

$$dq_{\mathrm{b}} = \frac{(1+0)}{1 \cdot 2} = 0.5 \tag{4.2}$$

$$dq_{\mathrm{ab}} = \frac{(1+0.1)(1+0)}{1 \cdot 2} = 1.05 \tag{4.3}$$

Table 4.1: An example for a TemporalElement containing two attributes/channels (a and b) with two data quality problems (i=invalid value and m=missing value).

Tempural Element	a	b	a.i	a.m	b.i	b.m
1	1.20	2.20	0.00	0.00	0.20	0.00
2	1.40	2.30	0.00	0.00	0.10	0.30
3	1.20	2.40	0.00	0.50	0.30	0.20
4	1.20	2.30	1.00	0.10	1.00	0.00
5	1.50	2.10	0.50	0.00	0.00	0.00

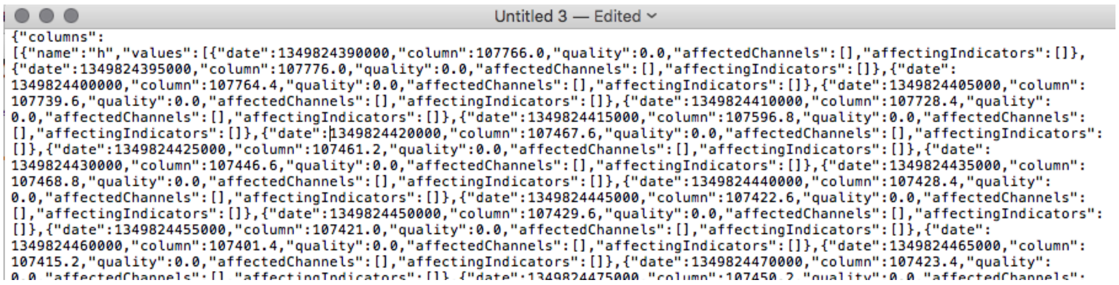
For the rest of this method I provide pseudo code in order to make it easier to follow and understand the individual processing steps, which is shown in Algorithm 4.1

The result of this is a nested Java object *TemporalData* that is automatically serialized into a JSON response messages using Dropwizard. Figure 4.5 shows an example of such an JSON response message.

This *TemporalData* consists of *TemporalColumn* elements, which again consist of *TemporalValue* elements. These classes as well as their dependencies are depicted in Figure 4.3. From the dependencies it should become clear that the object that carries the actual payload (i.e., the calculated values and the data quality) is the *TemporalValue*.

Algorithm 4.1: Read Aggregated Data.

```
Input: A list of columns, a list of quality indicators to be considered, a granularity,
       an aggregated dataset
Output: A temporal object with calculated quality measures
1  $temporalColumn \leftarrow newTemporalColumn()$;
2  set name of $temporalClumn$ depending on the selected columns;
3  while the iterator has a temporal object left do
4      get current temporal object from iterator;
5      get current $date$ from corresponding $temporalElement$iterator;
6      $totalColumnQuality$;
7      $affectingColumns$;
8      for each column of the columns do
9          $totalIndicatorQuality$;
10         $affectingIndicators$;
11         for each selected indicator do
12             get $indicatorQuality$ of the current temporal object and column;
13             add the $indicatorQuality$ to the $totalIndicatorQuality$;
14             if indicator has highest number of deficiencies then
15                 set $indicator$ as the most impacting;
16             end
17             if indicatorQuality is greater than 0 and indicator not in
               $affectingIndicators$ then
18                 add $indicator$ to $affectinIndicators$;
19             end
20         end
21         calculate the $columnQuality$ by using the mean of the
           $totalIndicatorQuality$;
22         add the $columnQuality$ to the $totalColumnQuality$;
23         if column (with all indicators) has the highest number of deficiencies then
24             set $column$ as the most impacting;
25         end
26         if $columnQuality > 0$ and column not in $affectingColumns$ then
27             add $column$ to $affectinColumns$;
28         end
29     end
30     $affectedColumns \leftarrow mostImpactingColumn$ and $mostImpactingIndicator$;
31     $temporalObjectQuality \leftarrow$ mean of $totalColumnQuality$;
32     $temporalObjectValue \leftarrow$ mean of $totalColumnValues$;
33     $temporalValue \leftarrow$ new TemporalValue($date$, $temporalObjectValue$,
       $temporalObjectQuality$, $affectingColumns$, $affectingIndicators$);
34     add $temporalValue$ to $temporalColumn$;
35 end
36 return $\vec{x}^{(k)}$;
```


Untitled 3 — Edited

```
{"columns":
[{"name":"h","values":[{"date":1349824390000,"column":107766.0,"quality":0.0,"affectedChannels":[],"affectingIndicators":[]},
{"date":1349824395000,"column":107776.0,"quality":0.0,"affectedChannels":[],"affectingIndicators":[]},{"date":
1349824400000,"column":107764.4,"quality":0.0,"affectedChannels":[],"affectingIndicators":[]},{"date":1349824405000,"column":
107739.6,"quality":0.0,"affectedChannels":[],"affectingIndicators":[]},{"date":1349824410000,"column":107728.4,"quality":
0.0,"affectedChannels":[],"affectingIndicators":[]},{"date":1349824415000,"column":107596.8,"quality":0.0,"affectedChannels":
[],"affectingIndicators":[]},{"date":1349824420000,"column":107467.6,"quality":0.0,"affectedChannels":[],"affectingIndicators":
[]},{"date":1349824425000,"column":107461.2,"quality":0.0,"affectedChannels":[],"affectingIndicators":[]},{"date":
1349824430000,"column":107446.6,"quality":0.0,"affectedChannels":[],"affectingIndicators":[]},{"date":1349824435000,"column":
107468.8,"quality":0.0,"affectedChannels":[],"affectingIndicators":[]},{"date":1349824440000,"column":107428.4,"quality":
0.0,"affectedChannels":[],"affectingIndicators":[]},{"date":1349824445000,"column":107422.6,"quality":0.0,"affectedChannels":
[],"affectingIndicators":[]},{"date":1349824450000,"column":107429.6,"quality":0.0,"affectedChannels":[],"affectingIndicators":
[]},{"date":1349824455000,"column":107421.0,"quality":0.0,"affectedChannels":[],"affectingIndicators":[]},{"date":
1349824460000,"column":107401.4,"quality":0.0,"affectedChannels":[],"affectingIndicators":[]},{"date":1349824465000,"column":
107415.2,"quality":0.0,"affectedChannels":[],"affectingIndicators":[]},{"date":1349824470000,"column":107423.4,"quality":
0.0,"affectedChannels":[],"affectingIndicators":[]},{"date":1349824475000,"column":107450.2,"quality":0.0,"affectedChannels":
```

Figure 4.5: An example of a serialized client response.

4.2 Implementation of the Visualization

After explaining how the data is loaded, aggregated and transformed (i.e.; serialized for sending it to the client), in this section I explain how I set up the proposed visualization, by using the libraries discovered in Chapter 3. For the visualization itself I used *d3.js*, a JavaScript library „for manipulating documents based on data“. Thus, throughout this section *d3.js* elements and code snippets are provided for each of the visualization problems faced. Figure 4.6 gives an first overview of the visualization containing both views, the *quality view* and the *detail view*.

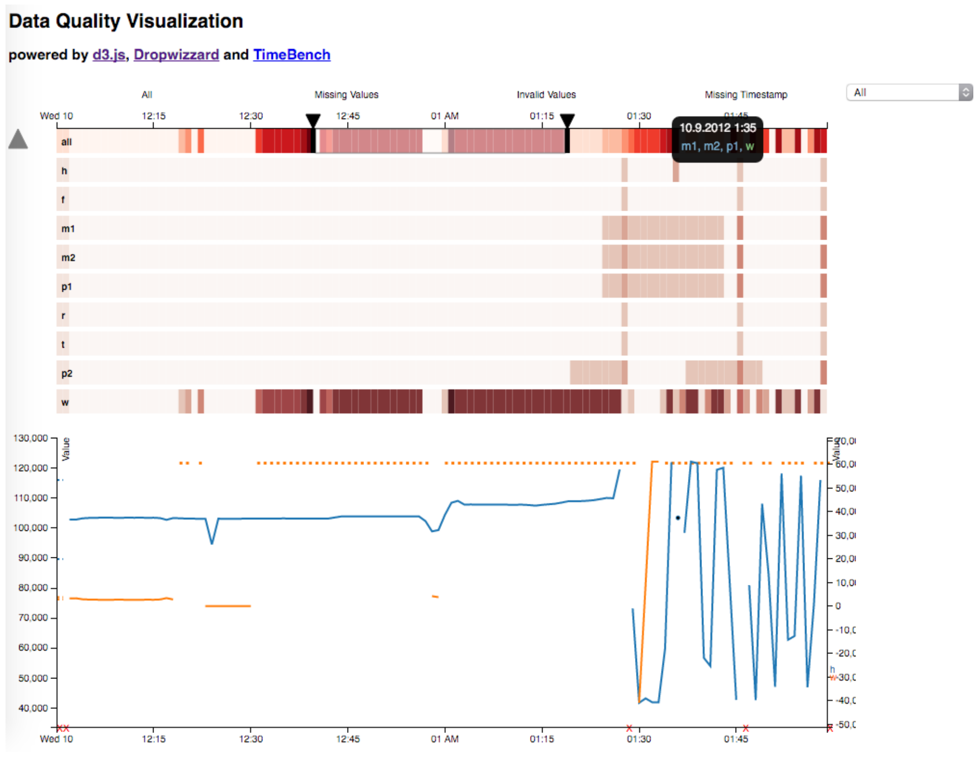


Figure 4.6: An overview of the implemented prototype and its data quality visualization.

4.2.1 Quality View

The quality view was nearly implemented according to the proposed design. The differences between the proposed design and the implemented solution are examined in the following. Figure 4.7 shows an overview of the quality view.

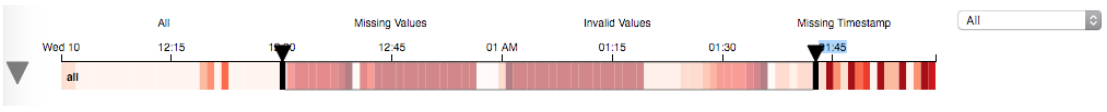


Figure 4.7: Overview of the quality view, displaying all data quality problems at the same time.

The Stripe

The stripes, which display data quality problems in overview, were implemented respectively to its design (compare with 3.2.1). Each of these stripes are drawn by using multiple *d3.js* rectangle elements, so called ticks that together compose one stripe. The size of such a rectangle element was calculated by using both, the size of the visualization and the number of temporal elements, given the chosen granularity of minutes. This granularity was determined during the expert interviews, as it best fits the provided data. Hence, the width of each tick was calculated by dividing the given width of the visualization by the number of temporal elements. The color of each tick was determined by its respective data quality and a pre-defined *d3.js* color scale. Different *d3.js* color scales were used for each quality indicator and the *all* stripe. Additionally, I used HSL colors with slightly decreased lightness values for all individual stripes in order to accentuate the *all* stripe while moving the individual stripes to the background. Listing 4.1 shows the code snippet of how the ticks are colored. The Figures (see 4.8, 4.9 and 4.10) show the different all stripes for the different quality indicators selected in the quality indicator drop down.

Listing 4.1: Color the Ticks of the Strip.

```
//update selection
ticks
   .attr("fill", function (d) {
      var color = d3.rgb(colorScale(1-d.quality));
      if(columnName!="all"){
         var colorHSL = color.hsl();
         colorHSL.s = colorHSL.s*0.5;
         color = colorHSL.rgb();
      }
      return
          "rgb("+color.r+","+color.g+","+color.b+")";
   });
```

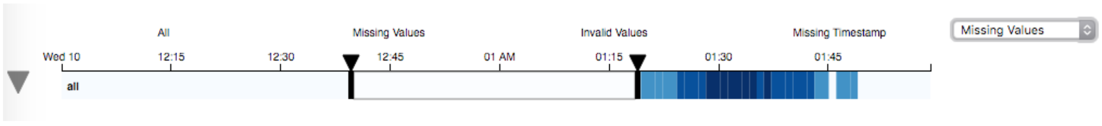


Figure 4.8: Quality view displaying missing values.

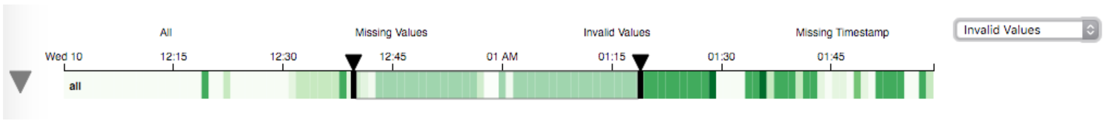


Figure 4.9: Quality view displaying invalid values.

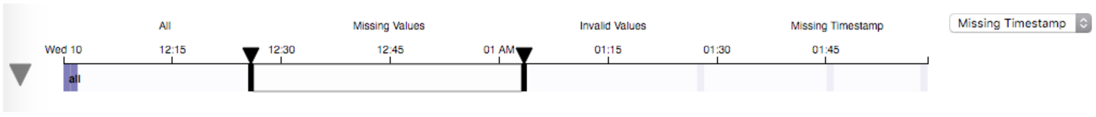


Figure 4.10: Quality view displaying missing time stamps.

The Toggle

The toggle was implemented to its corresponding design and drawn using a *d3.js* polygon element. Figure 4.11 shows the quality view with the expanded individual channels after the user clicked on the toggle. This interaction is handled by a standard *d3.js* on-click event handler.

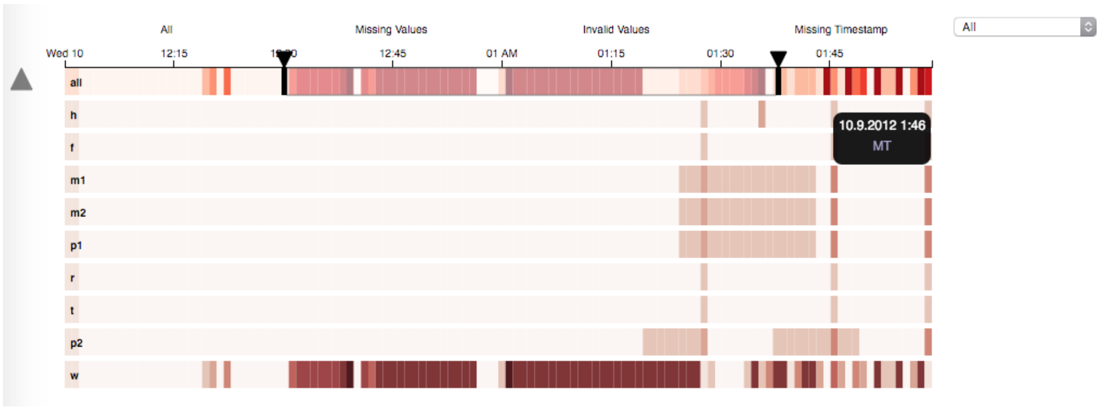


Figure 4.11: The quality view with expanded channels.

The Slider

The slider was first implemented according to its design, but later adapted during the expert interview sessions. One such adaption was that arrows were added on top of the range slider boundaries. Additionally, the scale of the data was added on top of the *all* stripe as well as the color highlight during a drag event was removed. The slider itself is composed of three *d3.js* rectangle elements for the range and the two boundaries, as well as two polygon elements for the indicating triangles on top of the boundaries. In order to add an additional scale I used a *d3.js* axis element, which I positioned before the *all* stripe. For the interaction I used standard *d3.js* on-click event handlers. Listing 4.12 shows the implemented time slider.

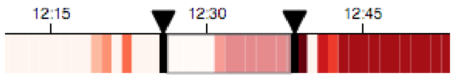


Figure 4.12: A time slider in the quality view for traversing time.

The Legend

Omitted during the expert interview sessions, since the meaning of most elements was apparent for the domain experts. However, if needed, the legend can be easily added and displayed in the future.

The Dropdown

The dropdown was implemented according to its design. During the expert interviews the concern was raised that it could be integrated better into the quality view. Thus, a second option was implemented, where the data quality indicators can be chosen by clicking on texts above the quality view. Figure 4.13 shows both options, the quality indicator dropdown and the mentioned textual list.

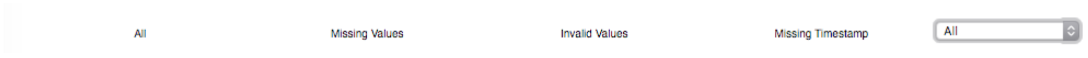
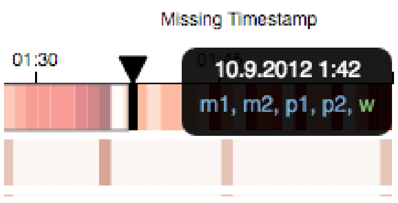


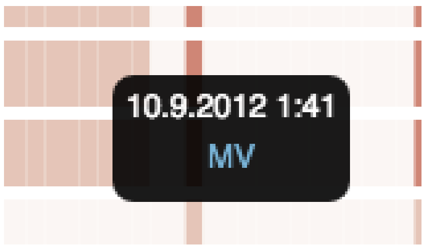
Figure 4.13: Dropdown menu for selecting the data quality problem in the quality and detail view.

The Tooltip

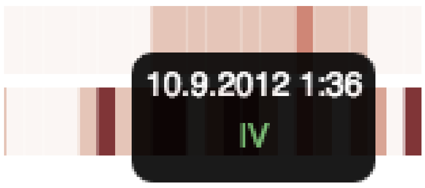
The tooltip was implemented to its respective design. Figure 4.14 shows the tooltip for the *all* stripe, which is shown whenever a user hovers over a certain tick. Furthermore, the Figures 4.8, 4.9 and 4.10) show the different tooltip variations depending on the channel and quality indicator combination. For drawing the tooltip I used a *d3.js* on-mouseover event handler, which shows a html division element with more information whenever a user hovers over a certain tick. The data for this division is simply pulled from the data bound to the respective tick.



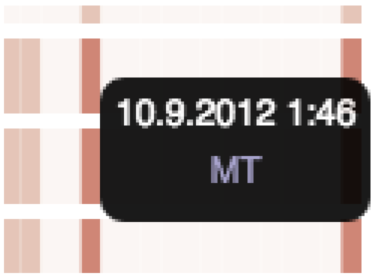
(a) all



(b) missing values



(c) invalid values



(d) missing timestamps

Figure 4.14: Tooltips depending on the combination of channels and quality-indicators.

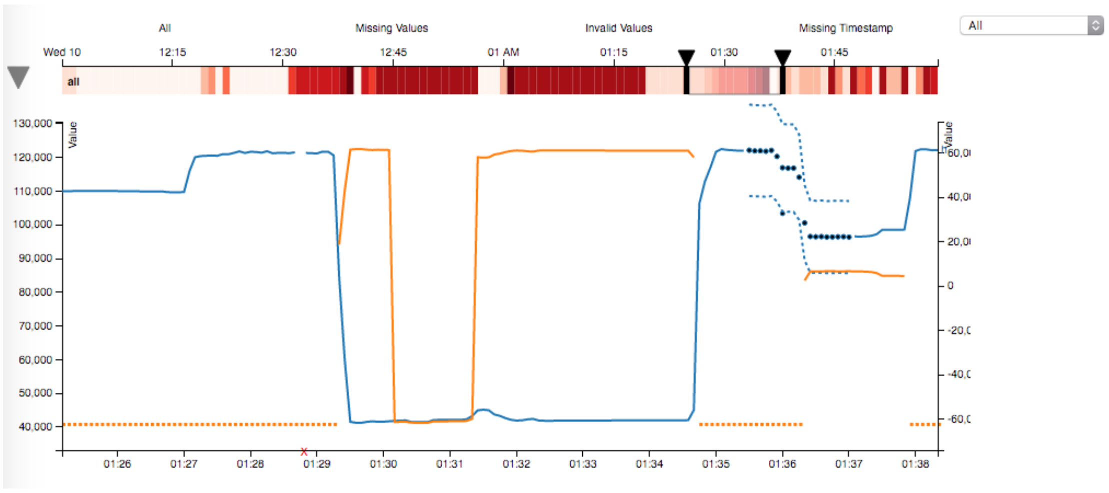
4.2.2 Detail View

The *detail view* was implemented with minor adaptions, which are described in the following paragraphs. Figure 4.15 shows an overview of the implemented detail view.

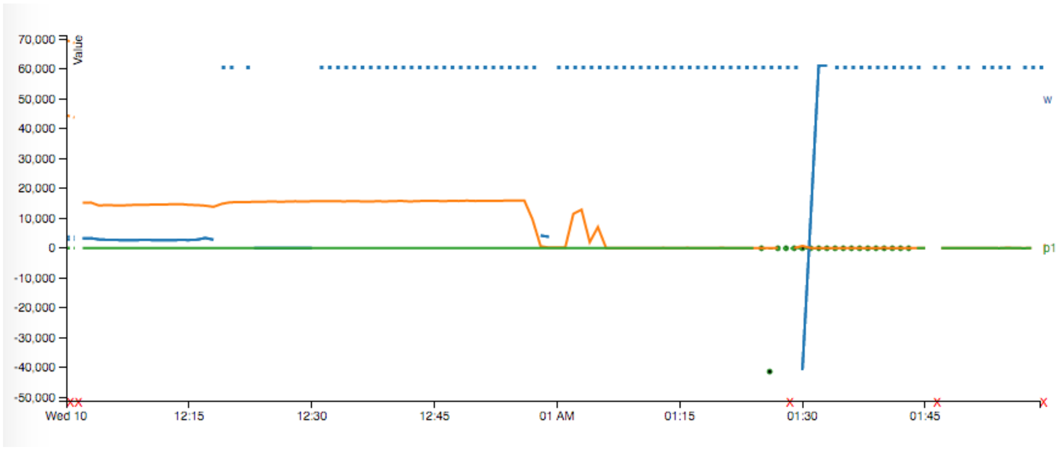
General

- *Number of Channels:* one outcome of the expert interviews was the decision to let the user decide how many channels he or she wants to add to the detail view. The result of this was to omit the initially proposed maximum of 2 channels in favor of the possibility to display one or multiple channels at the same time.

- *Dual Axes:* a subsequent question of having the possibility to add multiple channel is how or whether to implement a dual axis. In the expert interviews it was concluded that the axes should adapt to the number of channels. This means that a dual axis should be displayed if two channels were added and only one axis if one or multiple channels were added. Another question that was raised when using two axes was how many *d3.js* domains to use. For example, when I first implemented the detail view I only used one domain, which was scaled according to the channels that were added to the detail view. In a second iteration I changed this implementation, to use two different domains whenever two channels are added to the detail view. Figure 4.15a shows the detail view with such dual axes. However, as soon as a third channel is added all the three displayed channels again share



(a) with a dual axis



(b) without a dual axis

Figure 4.15: Overview of the detail view.

only one domain, which then again is scaled to all the contained channels. The second axis was added and removed accordingly. Figure 4.15b shows the detail view with more than two channels, but with only one single axis, which presents one single domain that is used in the background.

- *Domain:* when adding the data quality problems to the data detail view, I noticed that the visualization exceeded the natural domain limits of the data. This problem was discussed during the expert interview sessions. The solution was to expand the domain by 10-20%. For this, I subtracted the minimum value of the domain from the maximum value of the domain. The resulting geometric distance was then multiplied by 0.1 in order to obtain the desired 10%, which were then added to the domain (i.e., maximum + distance as well as minimum - distance).

Missing Values

In order to visualize missing values I first had to manually add them to the processed data set by changing the zeros in the „[channel].missingData“ column of chosen channels to a one. Furthermore, I also manually added imputed values for the channels and values chosen before, in order to be able to display those missing values. This way also the imputed values were loaded and aggregated by *TimeBench* and sent back to the client with the response. I then used two approaches to visually display these imputed missing values. First, I used *circles* in order to display the imputed values added before, by binding the imputed values to *circle* svg elements using *d3.js*. This way I already had circles displayed in the *detail view*. The second approach was to use two svg *line* elements that I used to display upper and lower confidence bands [Dar15]. Those confidence bands where used to indicate uncertainty. Figure 4.16 shows the combination of these circles with the confidence bands. Listing 4.2 shows the most important lines of code for creating the upper and lower confidence bands.

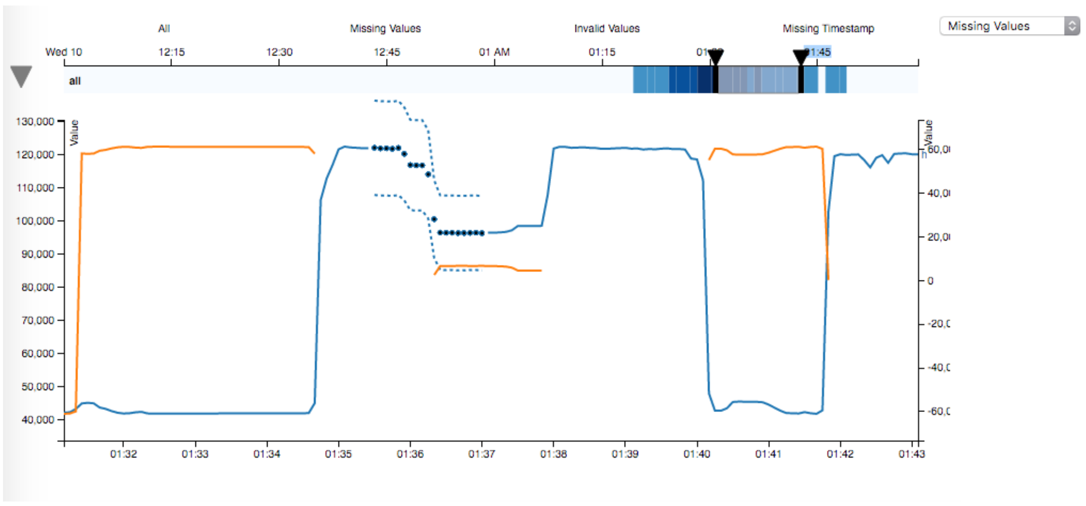


Figure 4.16: Implemented design option for *missing values* in the detail view.

Missing Time Stamps

The initial design options for missing time stamps were derived from the options for missing values. Therefore, for the prototype I chose the second design option for missing values, which is to indicate missing timestamps with a red „X“ along the x-axis. Figure 4.17 shows two missing timestamps. In *d3.js* I simply iterated over all missing timestamps and append a svg text element for each. I then us the pre-defined height of the detail view to place the respective missing time stamp along the x-axis.

Listing 4.2: Function to calculate the confidence interval

```
//the confidence interval for each value is calculated
//when loading the data
channels.forEach(function(channel){
   var n = channel.values.length;
   channel.values.forEach(function(d){
      d["confidence"] = se95(d.column,n);
   });
});
...
//using a svg line element
lineUpperBoundaryLeft = d3.svg.line()
   .defined(definedFunctionForMissingValues)
   .x(function(d) { return X(d); })
   .y(function(d) {return yLeftScale(d.column +
      d.confidence);}),
...
//the function to calculate the 95 confidence interval
   value.
function se95(p, n) {
   return Math.sqrt(Math.abs(p*(1-p)/n)*1.96);
};
```

Invalid Values

Even though four different options were proposed in the design chapter, none of them were fully implemented. Nevertheless, parts of these approaches were combined in order to implement data quality indicators for invalid values in the detail view. In general the final implementation is a mixture between option two and four of the initially proposed design options (see Section 3.2.2). This means an invalid value is not displayed at its original position, but along the closest extreme value. Hence, the invalid value will be either displayed as a maximum or minimum of the currently shown fraction of the respective channel. This directly exploits the idea of option two and four to indicate invalid values at different positions rather than emphasizing the invalidity on the same spot. Figure 4.18 shows a dashed line along the maximum value of the orange channel *w*. In general, I used the *d3.max* and *d3.min* function on the *domain* of all filtered valid values in order to determine the maximum and minimum of each channel when loading the data. For displaying the values I simply iterated over all invalid values and calculated their distance to either extreme value. Based on this calculated distance I either display the invalid value as a maximum or minimum in the detail view.

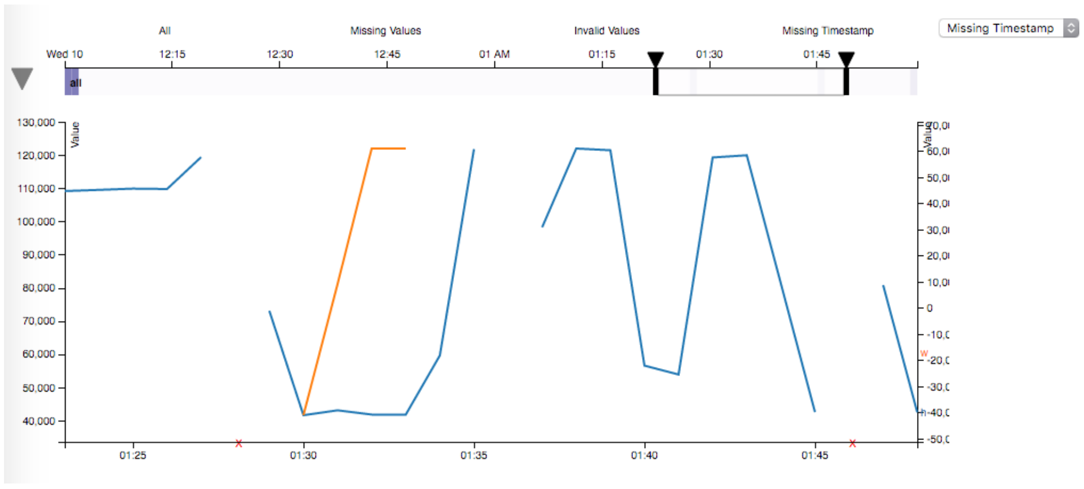


Figure 4.17: Implemented design option for *missing timestamps* in the detail view.

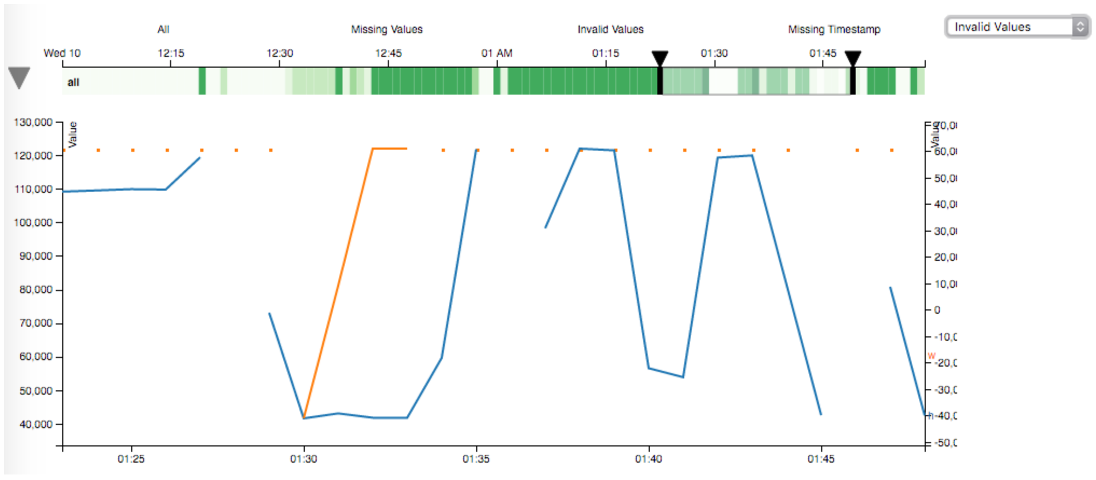


Figure 4.18: Invalid values, displayed in the detail view.

CHAPTER 5

Discussion

In this chapter I examine the selected evaluation approach and give rationales for the derived evaluation method in Section 5.1. The key results of this evaluation are then reported and discussed in Section 5.2, where the focus lies on the differences between the proposed design and the implemented solution as well as on the improvements that were derived. In addition, „lessons learned“ from the implementation are provided in Section 5.3. In Section 5.4 the technical and visual strengths and limitations of the implemented prototype are presented and recommendations for future work are made.

5.1 Background Information

5.1.1 Problem, Research Gap and Approach

In the introduction of this work I described why data quality is important in order to take well-informed decisions and investigated whether the visualization of data quality problems improved the decision making process. During this investigation I discovered several references [SZ12, ZSC07, WXYR11], which showed that data quality visualization can help decision makers to increase awareness of the accuracy and completeness of the data as well as it improves the decisions made.

Based on this finding, a literature research was conducted in order to find information visualizations that integrate data quality. This research yielded that only few comparable solutions exist, which deal with data quality problems in combination with multivariate and time-oriented data.

In order to address this research gap I derived and proposed several design options for visualizing data quality problems, which were then implemented and evaluated in a prototype (see Chapter 4).

5.1.2 Evaluation Methods

For the purpose of evaluating this prototype I partly used a „usage scenario“, as proposed by Isenberg et al. [IIC$^+$13]. I additionally examined papers on evaluating information visualizations and discovered the contribution of Lam et al. [LBI$^+$11], which provides „Seven guiding Scenarios for Information Visualization Evaluation“. These scenarios are a combination of evaluation goals, outputs, questions and methods. From these possible scenarios I identified Scenario 6 „Evaluating User Experience“ and its goal as the one that best matches the research questions of this work. „to understand to what extent the visualization supports the intended tasks as seen from the participants' eyes and to probe for requirements and needs. [IIC$^+$13]“ Scenario 6 is described as „specific to a given design problem“ and „focuses on a specific visualization“, both of which characteristics match the introduced research problem. Derived from this scenario I explored the proposed evaluation methods and selected *informal evaluation* as an extension to the initially used *usage scenario.* According to the authors, informal evaluation is conducted by showing a functional prototype to domain experts, while letting them play with and observe typical system features. This way questions about intuitiveness and functionality are answered. Additionally, it identifies design flaws and improves the implementation of the proposed ideas.

The main argument for using informal evaluation was provided by Isenberg et al. [IIC$^+$13], who discovered that QRI (Qualitative Result Inspection) is the most widely used form of evaluation for information visualizations, accounting for 46% of all evaluation scenarios used. QRI in its essence is an evaluation through qualitative discussions and assessments of visualization results and was added as new category to the usage scenarios introduced by Lam et al. [LBI$^+$11].

To sum up, I argue that a formative and qualitative evaluation is the most appropriate approach for the research problem introduced, where the main question lies in whether the proposed information visualization can be improved.

5.1.3 Evaluation

In order to obtain the key results I first evaluated the design choices made by means of immediate evaluation (i.e., comparing it to guidelines, design principles, or prior contributions) in Chapter 3. Additionally, I conducted semi-structured interview sessions with visualization experts throughout the design and the prototyping stage, which took place four times during the design stage and twice per milestone (i.e., prototype, mvp, final release). The structure for these sessions was provided by the features, which were implemented for the respective milestone. For instance, in one of the first sessions the data quality view was demoed at the beginning and the group discussed the effectiveness of all its individual elements. The main purpose of these sessions was to either confirm, refine, or reject the design decisions (i.e., reflection) made in both stages. In these sessions up to 4 different experts participated in the discussions. All experts were male, between 25 and 35 years old and researchers in the area of information visualization. Two of these experts already worked with partners in the drilling industry, which faced similar problems to the one identified in this work. Moreover, in the design stage I covered the visual encoding and interaction design space, which is based on human perception and cognition. In the implementation stage I looked at whether the visualization has achieved its design goals and compared the prototype with current state-of-the-art systems as well as techniques [LBI+11]. Notwithstanding, I did not cover the pre-design, deployment and re-assessment stage, as they have either already been covered in prior work [ABG+14], or were not in scope of this work. The results of the evaluation can be found in Section 5.2.

5.1.4 Research Questions

In the following I provide a linkage between the research questions of this work (see Section 1.3) and the answers given to them in this chapter.

- Question 1 is primarily answered in Section 5.2.1
- Question 2 is primarily answered in Section 5.2.2
- Question 3 is primarily answered in Section 5.4

5.2 Key Results

The results obtained during the expert interview sessions are presented in the following. For the purpose of comparability I divided the discussion of the key results according to Chapter 3 and 4 into one section for the *quality view* and one for the *detail view.*

5.2.1 Quality View: Visualizing Data Quality in Overview

Toggle buttons, Drag-and-Drop and Tooltips

Following the proposed design, a toggle button was implemented that shows and hides the underlying individual channels below the „all“ stripe as well as to make efficient use of space for the quality view. Furthermore, common design elements such as drag-and-drop and tooltips were used for providing interactivity and additional information. The decision to use these three elements were based on *Jakob's law.* The main idea of this law is that the user's expectations for most design and interactivity elements are based on what is common [Nie00, Saf10]. This means that the law essentially promotes the use of standardized elements at common places in order to facilitate an intuitive display.

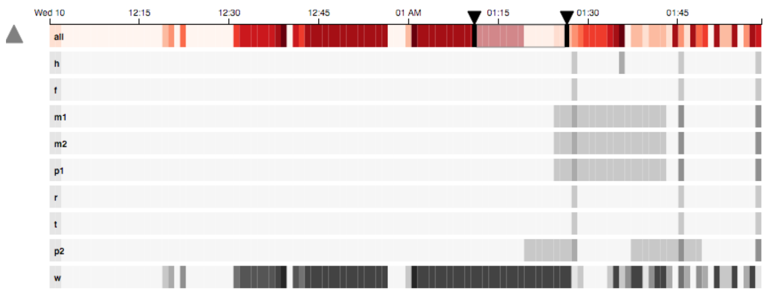
Nonetheless, one proposed standard design element was questioned during the expert interviews. The experts specifically raised concerns about whether a dropdown menu is sufficient for selecting the quality indicators displayed in the detail view. Their first feedback was that they did not perceive the dropdown menu as integrated in the quality view. As a result, they worried that without further guidance the user might be confused about the function and usage of the dropdown. Based on this feedback another approach was proposed and implemented to better integrate the selection of quality indicators into the visualization. This approach provided single quality indicators listed as texts above the stripes (see Figure 4.7). This way, the user can select multiple quality indicators at the same time, which was not possible with a dropdown menu, as it only allows to select and display one at a time. However, this approach proved not to be scalable, since adding additional quality indicators would soon lead to a lack of space, and as a consequence the dropdown menu was favored over the list.

Quality Levels, Color Coding and Emphasize

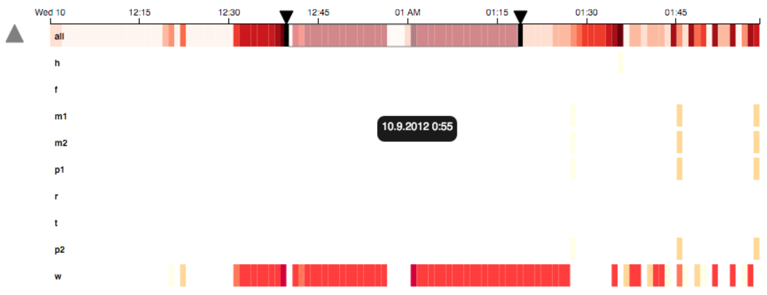
Another question that occurred during the interview sessions was how to differentiate between the „all“ stripe and the individual stripes for the channel, considering different quality levels, which already make us of different shades. One immediate solution was to use color coding [Chr75]. However, since color coding was already utilized in combination with different quality indicators, it was not immediately clear how to exactly apply color coding. Moreover, with using the same shades for both, the „all“ stripe as well as for the individual stripe the experts had problems to see the difference between the two. Therefore, I proposed three options to color code the quality levels in the individual stripes (see Figure 5.1) in order to avoid confusion. In the first option gray scales were used to differentiate the individual stripes from the „all“ stripe (see Figure 5.1a). In the second option d3.js's built-in *brighter* and *darker* „rgb()“ function were utilized (see Figure 5.1b) in order to set the individual stripes apart from the „all“ stripe. In the third option a saturation of only 25 percent was applied in order to create distinguishable shades (see Figure 5.1c) for the data points of the individual stripes, according to their quality levels. The subsequent feedback for these three options yielded that the second option (i.e., brighter/darker) is not ideal, since the user might miss the brighter spots, and therefore overlooks differences of the data's quality. Even though the remaining options were both considered as good choice, the experts favored option three for its clear visual distinction between the different data quality levels as well as between the „all“ stripe and the stripes for the individual channels.

Fisheye

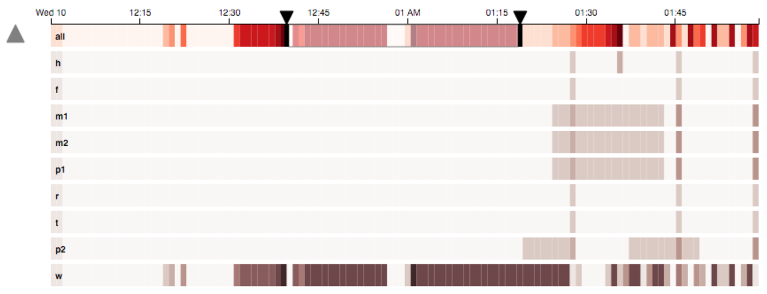
During the interview sessions one design option that was advocated was the fish-eye (see Figure 5.2). The fish eye is an visualization technique where the focus lies only on a minor part of the visualized data. This means that the parts before and after the area in focus are distorted, comparable to a fish-eye lens. However, during the discussions with the experts it was decided that for the prototype a static view of the data quality is preferable. The reason for this decision was that the intent of a fisheye visualization, to see local detail and global context, has already been provided by interactively linking the quality and the detail view. Another argument against the use of a fish eye visualization in the quality view was the fact that other contributions have shown that it can also cause confusion [SB92].



(a) color coded and grey scale applied



(b) color coded with d3.js's function „brighter“



(c) color coded with 25 percent of saturation

Figure 5.1: Color coding options for quality levels in the quality view.

Slider

One of the first critiques during the expert interview sessions was that the click-able boundaries of the range slider are not obvious to the user. As a consequence, together with the experts means to additionally indicate these boundaries were discussed and sketches were created. The result of this was to add arrow icons on top of the boundaries in order to suggest clickability [Oka15, Lor15] and to also make the boundaries and the range slider more intuitive. Figure 5.3 shows the concept drawing for these modifications.

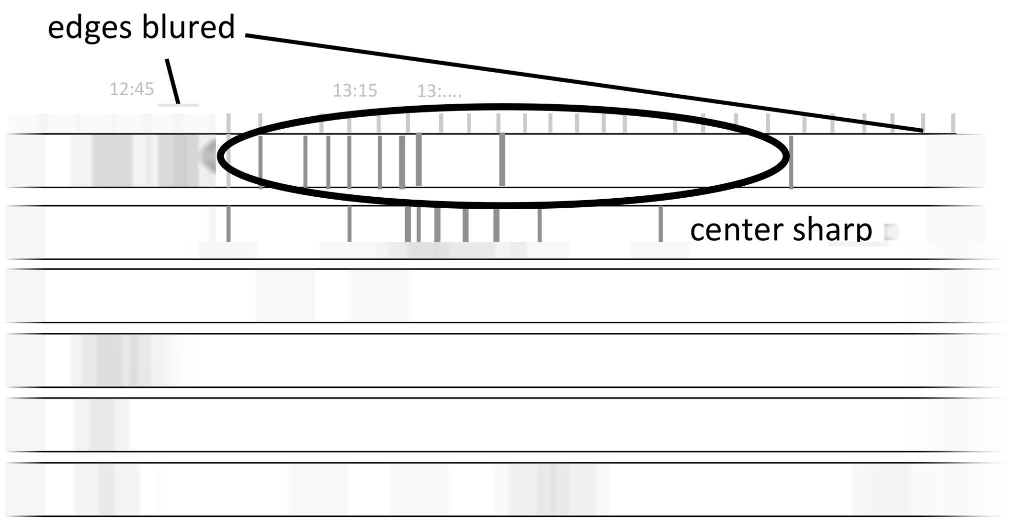


Figure 5.2: Sketch of a technique called fish eye that visualizes data quality in overview.

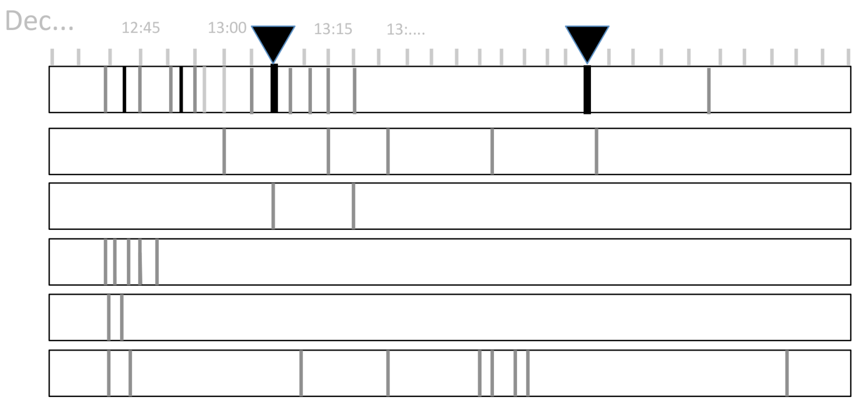


Figure 5.3: Improvement for the slider suggested during the expert interviews.

Legend

One finding of the evaluation was that the legend did not provide the user with additional value, and thus was omitted from the visualization. The reasoning for the removal was that all elements are self-explanatory and that a legend then only requires more space, while not providing added information. However, in order to differentiate the individual channels it was necessary to label the quality stripes using the respective names of the channels. And if needed the legend can easily be added and displayed in the future.

5.2.2 Detail View: Visualizing Data Quality in Detail

After implementing the majority of features during the first sprint to reach the first milestone (i.e., „Prototype“), I started the second sprint to create a Minimum Viable Product (MVP) (i.e., milestone two). The main concern of this sprint was to explore visualization approaches in detail for the three data quality problems (i.e., missing values, invalid values and outliers) identified. The resulting selected and implemented visualization approaches were evaluated by means of expert discussion sessions, which will be examined in the following.

Missing Values

Before starting the implementation for visualizing missing values a discussion session was conducted with the experts. During this discussion session the four different visualization options that were initially proposed in the design stage were reviewed. During this review it was concluded that the first option, to use bars in the background, can be omitted due to its impracticality. The argument for omitting it was that the bars might not easily be displayed on different zoom levels. Additionally, it was assumed that using bars may lead to a more cluttered visualization, compared to the other proposed design options. The experts also raised the concern that boxes do not provide the user with additional information.

As a consequence, other visualization options such as displaying imputed values, where imputed values are visualized as separate dots, where discussed. This discussions showed that using separate dots leads to wrong interpretations. For instance, it was noted by the experts that without extra knowledge the separate dots might be perceived as important data values instead of missing values. Hence, it was mandatory to provide some indication of uncertainty for the user to not confuse these missing values with more important valid values. For this, the experts suggested to utilize confidence bands, which indicate the margin of deviation and by that prevent the user from misinterpreting the importance of the visualized missing values. An extrinsic and coincident integration of uncertainty was chosen, because the complexity of the task is rather low, which according to scientific evidence [Mac15] suggests that it enables the user to interpret both, the data and uncertainty more accurately. In a next step different design options for these confidence bands were investigated, where the first option was to color code the complete section of the confidence band. However, this option was immediately rejected, since the same arguments as for boxes apply, rendering it impractical on different zoom levels. The experts also pointed out that by color coding the whole section of the confidence band the missing values would be emphasized once more, rather than being visually differentiated from valid values. Furthermore, for the confidence band itself a dashed rather than a continuous line was considered. The reason for this decision was that by the experts a dashed line was perceived to accentuate uncertainty (i.e., the lack of information). Another outcome of these discussions was that the imputed data points should be visually differentiated to the rest of the values. Therefore, black as a neutral color was used for the dots of these imputed values. A last point of discussion was the color of the confidence band, where it was generally accepted that the dashed line should have the same color as the channel in order to associate the confidence band with the respective channel.

As a result of these discussions, all proposed options made in the design chapter were rejected and a fourth option was advocated that combined several ideas. Moreover, this option partly utilized ideas from the design options proposed for visualizing outliers. Figure 5.4 shows the sketch of this added approach, which can be compared with the implemented approach shown in Figure 4.16.

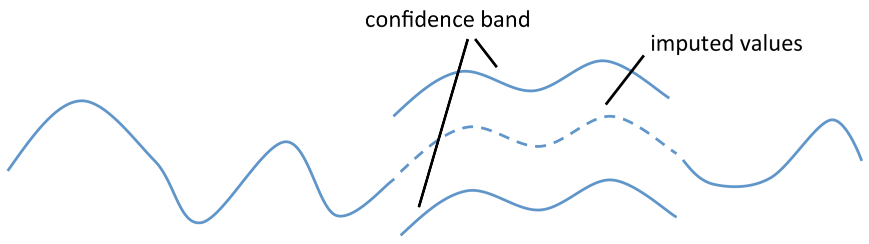


Figure 5.4: Fourth proposed design option for missing values in the detail view.

Missing Timestamps

In addition to missing values, missing time stamps were discussed during the expert interviews. One of the findings of the interview was that both data quality problems are similar in nature, which is why similar options to the ones of missing values were considered. For example, one option was to use a dashed box in the background of the detail view, similar to the boxes proposed for missing values. However, it was immediately concluded that the counter arguments for not using boxes for missing values also apply for missing time stamps. As a result of further discussion, it was decided to use the axes in order to indicate missing timestamps rather than directly in the graph. The reasoning for this was that this way it would clearly differentiate the visualization of missing values and missing timestamps. The final implemented choice was to use an „X“ sign on the x-axis in order to indicate missing timestamps.

Invalid Values

In another interview session visualization approaches for invalid values were evaluated. For this, together with the experts, the visualization options proposed in Chapter 3 were evaluated and potential pitfalls as well as prospective enhancements were discussed. For example, considerations to use option one (i.e., to use color coding) yielded that it misleads the user, as he or she then might interpret the color coded value of higher importance than others. As a result, design option one (i.e., to color code invalid values) and three (i.e., to substitute invalid values with imputed values) were rejected, as they both make use of color coding. Figure 5.6 shows sketches that were created during the discussion.

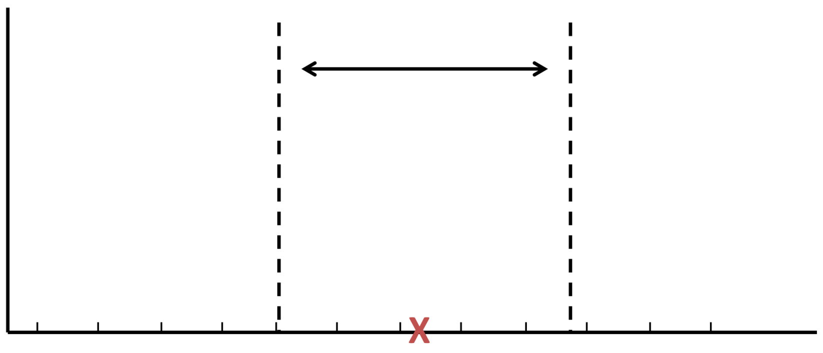


Figure 5.5: Design of the proposed visualization approach for missing timestamps.

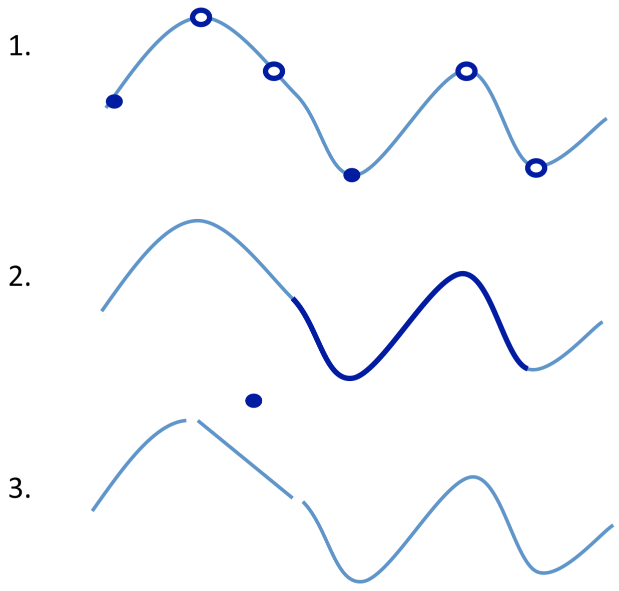


Figure 5.6: Different ways suggested during the expert interviews to indicate invalid values.

In the following I examined design option two in more detail. The idea of this design option was to only indicate invalid values, rather than visualize it, similar to option four. The reason for this is the nature of the provided data. In our business case the experts characterized an invalid value as an entry that is out of a defined range (i.e., either higher or lower than a defined threshold). Consequently, by only indicating their occurrences rather than visualizing them the scale of the visualized values is not affected, and thus a more accurate visualization can be provided.

The following question was then to decide whether to indicate invalid values directly in the graph (design option two) or to indicate them on the axes (design option four). Since the axes were already used to indicate missing timestamps, design option two was favored over design option four. During further discussions regarding option two the idea arose to use rectangles in order to indicate invalid values in the graph. Together with the experts it was also determined that the rectangles should be vertically positioned at either the maximum or the minimum of the respective valid data displayed. The reason for assigning either the maximum or minimum to the y-position of such a invalid value was to not affect the scale of the data display, while still carrying the information of whether the invalid value was lower or higher than the defined threshold. Therefore, the vertical distance from an invalid value to the vertical maximum as well as the minimum of the valid data displayed was calculated. The y-position of the invalid value was then assigned accordingly, depending on whether the invalid value was closer to either the former or latter. The orange rectangles in Figure 4.18 show the final result of the proposed visualization.

Outliers

In the case of both visualization approaches discussed for missing values as well as for invalid values, I utilized ideas that were drawn from visualization options initially proposed for outliers. For instance, I used confidence bands for the visualization of missing values in order to express uncertainty. Additionally, most arguments made for the visualization approaches for invalid values also directly apply to outliers. One reason for this is that invalid values, as defined by the domain experts, can also be seen as outliers. As a consequence, it was decided not to visualize outliers, particularly since the focus of the use case was on missing and invalid data.

5.3 Lessons Learned from the Implementation

5.3.1 No single solution, higher level vs. lower level libraries

The first lesson learned was that only a few frameworks and libraries exist that cover all aspects from loading to visualizing multivariate time-oriented data. In addition, I spent a lot of time on investigating higher level libraries (i.e., built on top of d3.js), which at the end did not yield any potential results. Hence, I concluded that no single solution exists, and thus combined several lower-level frameworks and libraries in order to create the architecture for the prototype. This combination was one of the factors that slowed down the implementation. For instance, the necessary transformation of the loaded data in TimeBench to a format appropriate for d3.js was one of the resulting insufficiencies. On the other hand, utilizing lower level frameworks and libraries provided the most flexibility, which was particularly useful when implementing interaction capabilities between individual views.

5.3.2 References and examples regarding the architecture

After finding sufficient building blocks for the architecture I further learned that finding introductory examples for the frameworks and libraries chosen was easy, while finding reference material for a more advanced solution was substantially harder. For example, at the end my single resource for building the architecture were two quite outdated online articles about reusable visualizations with d3.js [Bos12, Pen13]. These articles particularly contained more advanced guidance on how to develop reusable charts, including considerations such as repeatability, modifiability, reconfigurability and extensibility.

5.3.3 Lack of documentation slows down the implementation

Another factor that impeded the implementation was the lack of documentation for TimeBench. As a result, the initial effort for loading the data was higher than initially expected as well as compared to prior experiences. Overall, a sufficient documentation for the libraries used proved to extensively promote a faster implementation of the prototype. Therefore, I particularly suggest to thoroughly consider the available documentation as an important criterion for selecting frameworks and libraries to assemble the architecture for an information visualization.

5.3.4 Visualizing multiple data quality problems is challenging

Besides the technical aspects mentioned, I also learned that visualizing multiple data quality problems in the same view and at the same time proved to be challenging. For instance, using color coding for one data quality problem, makes it insufficient to use the same technique for another. As a consequence, I had to constantly consider all the different data quality problems when exploring visualization options for one individual data quality problem. This was necessary in order to prevent applying the same or contradictory techniques to two individual data quality problems. And I particularly learned that only a limited number of channels exist, which offer different depths such as size, color or saturation. For example, when I investigated ways to display missing time stamps I had already proposed to visualize missing values in the graph, making it insufficient to visualize similar problems such as missing time stamps also in the graph. As a result, I indicated missing timestamps on the axes rather than visualize them together with missing values in the graph. Thus, I learned to use an holistic approach, which considers all data quality problems when selecting design options for individual data quality problems.

5.3.5 What doesn’t work for one data quality problem, does often not work for others

It often occurred that whenever a visualization option was rejected for one particular data quality problem it was usually also rejected for others. For example, one initially proposed design option was to display a bar where missing values or where missing time stamps occurred. In this case all the counter arguments raised for using bars for one problem also held true for the other. Hence, I concluded that when a design option was a poor choice for one specific data quality problem it was usually also a bad choice for others.

5.3.6 Being careful with visual emphasis

Another important lesson that I learned during the design and implementation stage was that emphasizing visualization techniques should only be applied with care. For example, when I proposed to use coding to visually emphasize invalid values, the experts assumed that these emphasized data points are special, rather than missing or invalid values. From this, I learned that such visual emphasizing techniques have the potential to mislead the user and concluded to only use them with great care (e.g.,after successful user testing). On the contrary, the evaluation also showed that not visually separating elements can also have a negative impact on the effectiveness of such visualizations. For instance, I first used the same colors and shades for both, the *all* stripe and all the individual stripes for the different channels. As a consequence, the experts did not immediately see and understand the difference between these two different stripes, leading to a necessary visual distinction.

5.3.7 Less is more

In the first iteration it was technically allowed to display all channels at the same time in the detail view. When this approach was evaluated the experts immediately raised the concern that displaying multiple channels at the same time would take away necessary space for visualizing data quality problems. These problems also emerged in connection with adding and removing channels from the quality view. For instance, when more than 15 channels are displayed in the quality view it would be hard to add one of these channels to the detail view, since the detail view would then be pushed out of view. I learned that too many details take away too much space for visualizing and conveying the most relevant aspects of data quality problems and might even lead to confusion of the user. Therefore, scalability should always be kept in mind when designing visualizations and I advocate to reconcile the number of details (i.e., channels, data quality problems) with the user in order to prevent visual clutter.

5.4 Strengths, Limitations and Future Outlook

5.4.1 State-of-the-Art

Overall, I examined different ways to visualize data quality problems in detail and overview, which only a few prior contributions have done in depth before. Derived from this I then suggested several novel visual design options for data quality problems in combination with multivariate and time-oriented data. The evaluation of these options largely concentrated on selecting the best design option proposed in the design stage as well as on further improving these selected options by means of qualitative discussions.

In this context it should be noted that no other contribution has been found, which explicitly provides design options for multivariate and time-oriented data. In addition to these design options, all the rationales discussed in the expert interviews (see Section 5.2) can be seen as another contribution of this work, building the grounds for future work.

5.4.2 Algorithms and Architectures

In addition to reviewing ways to visualize data quality problems, I also examined selected algorithms for the prototype implementation. Although numerous contributions were found that provide algorithms in connection with data quality problems, only a few of them make concrete suggestions of which algorithm to use for which data quality problem. One reason for this may be that such algorithms depend on the particular problem at hand, as argued by authors. In order to address this lack of guidance, I provided suggestions for algorithms in conjunction with certain data quality problems, such as using imputation in combination with missing values. However, one field where I could not find any resources was data quality aggregation. Besides my discussion of data quality aggregation in Chapter 4 I advocate to further explore algorithms for data quality aggregation in future work. For instance, it could be examined how using the median, instead of the algorithmic mean, to aggregate the data quality problems affects the effectiveness of the visualization.

The prototype contains most data quality aspects of Ward et al. [WXYR11], except for quality abstraction. To address this, means could be included that support the user during the choice of the underlying algorithm (e.g, self organizing maps, maximum likelihood, multiple imputation, nearest neighbour, etc.). For instance, by providing options for selecting the right algorithm for imputing missing values (e.g., like XGOBI or MANET [Hua05, TAF12]).

Furthermore, only a few contributions have offered technical details of their proposed designs [ZWRS07, Hua05, WXYR11]. Hence, in order to address this lack of information this work gives insights on how to deal with technical challenges when visualizing data quality problems. For instance, when loading, aggregating or transferring the data and its quality from the server to the client.

5.4.3 Scalability and Generalizeability

The evaluation of the prototype indicates that the selected strategy works well with the provided data sets, as the loading is performed within an acceptable time. In this regard it should be noted that no other data sets have been used, which means that no statement can be made regarding the scalability and generalizability of the prototype. Therefore, investigation of techniques to address scaling issues regarding the load process (e.g.,aggregation on-demand rather than pre-aggregation) should be conducted. Regarding the visualization of the introduced data quality problems, the proposed approaches can be easily generalized, and thus applied to similar domain problems (e.g., water quality monitoring). For instance, it would be possible to only use the quality view combined with a geographical map, where time on the map can be traversed by using the slider of the quality view. However, it also has to be noted that a few proposed approaches have several scalability limitations. One good example for such was the conclusion that only a maximum of two channels should be displayed in the detail view at the same time, which is based on the assumption that a user would usually only compare two channels at a time. And this user would then remove or add channels on demand. The rationale for this was that with too many channels the graph was perceived as overloaded and cluttered. Therefore, I concluded that a certain maximum of data quality problems displayed at the same time in one view should be defined, which is acceptable for the user.

5.4.4 User Study

In addition to these technical challenges this contribution also provides further ideas and arguments for certain interaction techniques and elements in combination with data quality visualization. However, it has to be pointed out that I neither conducted a user study nor did I perform a technical benchmark (i.e., for the algorithms used). Both would ideally provide a quantitative evaluation of the visual and technical effectiveness of the implemented prototype. Finally, I claim that a user study would provide additional arguments for the proposed design choices and should be subject to further research.

CHAPTER 6

Summary

In the beginning of this work I showed that decision makers unknowingly base their decisions on imperfect data, which often leads to poor decisions and high costs. These costs steadily increase with today's ever growing data volumes. Therefore, I investigated ways to visualize data quality problems in order to make decision makers more aware of them as well as to improve the decision making process. Despite existing literature, proving that data quality visualization improves decision-making, it was found that only little research has been conducted in the field of univariate and multivariate data quality visualization. As a result, I used a practical business problem in the drilling industry dealing with multivariate time-series data in order to address parts of this identified research gap. The data provided by the partner company represents different activities that are performed during a drilling process. The main concern of the partner was to derive actions to improve the quality of the overall process, whereas the main goal of this work was to develop a novel approach for effectively visualizing the contained data quality problems in overview as well as in detail.

For structuring this work I combined ideas from the design-science paradigm [vAMPR04] with ideas of the nine-stage design study methodology[SMM12] (see Section 1.4) and used a four stage approach (i.e., learn, design, build & deploy, and reflect). According to the design study methodology I first conducted a state-of-the-art literature research (learn) where I gathered the necessary knowledge related to the domain problem. I then developed data abstractions, visual encodings, and interaction mechanisms in the following stage (design), for which I primarily used immediate evaluation in order to justify the design choices made. As a result, I implemented selected design options in a prototype (build & deploy), and evaluated (reflect) them by means of expert interview sessions, which were used to either confirm, refine, or reject the selected design options.

Following the DSM methodology I first examined different contributions related to the domain problem of this work and categorized them into three categories. The review of all the approaches in these three categories further verified the initially identified research gap for data quality visualization in combination with multivariate time-oriented data, since neither solution fully addressed the domain problem introduced in this work.

As a consequence, I also explored approaches for visualizing the identified data quality problems (i.e.; for missing values, missing time stamps, invalid values, and outliers). This yielded that only few resources covering the visualization of data quality problems exist, while numerous resources are available that explain algorithms for detecting such data quality problems. For instance, I only found two potential approaches for visualizing missing values in overview (*aggregation plot* and *matrix plot*), while did not find any approach to visualize missing time stamps or invalid values.

Based on the ideas gathered throughout the state-of-the-art research, a technical as well as visual design was derived, and explained in Chapter 3. For the technical design I qualitatively evaluated different libraries and decided to use TimeBench for loading and transforming the raw data (i.e., loading, aggregating,etc.), while deploying Dropwizard to expose this transformed data through a web service and selected d3.js for visualizing this data. For the visual design I proposed and discussed several design options to visualize the data quality problems identified. For example, I first advocated to separate the prototype into a quality view and a detail view in order to reduce cognitive load. For the quality view I advocated to use so called quality stripes in combination with a time slider in order to provide an overview of the data's quality. Regarding the detail view I proposed and discussed up to 4 different design options for displaying the data quality problems introduced in this work. Supplemental to proposing these design options, I investigated interaction techniques for linking the two views. Consequently, all the selected design options where implemented in the form of a prototype for the purpose of latter evaluating them.

During the implementation I first answered the question of how to set up and combine the libraries chosen in the design chapter. At the beginning of the implementation chapter I demonstrated how Dropwizard *Resource* classes were used as a basis for the prototype and showed how request types, provided to the client, were defined in these classes. I then explained the business logic included, and how the data and its quality is loaded, stored and aggregate by the use of a Data Access Object and the TimeBench library. Moreover, I provided an introduction into TimeBench's *TemporalDataset* and Dropwizard's *Representation Class* and explained how they are used to load and store the data. Derived from the data structures I investigated the problem of loading high volume data and suggested to use a combination of data aggregation and on-demand data slicing in order to address the problem of transferring high volume data from the server to the client. Generally, the proposed solution was choose the aggregation level of the data based on the selected time range, and by that reduce the volume and size of the transfered data. Besides the technical challenges, I examined how d3.js can be used to visually address the data quality problems introduced in this work. And covered aspects from visualizing the requested data and its data quality to handling the user's interaction between different views, while highlighting the differences between the proposed and implemented designs. The reasons for these difference where then discussed in the following Chapter 5.

In the discussion I first elaborated on why I decided to use a formative and qualitative evaluation method, and briefly explained how the evaluation was conducted. As a result of the key findings made during the evaluation, I reported the conclusions that were drawn from them. Additionally, I showed that some standardized elements (e.g.; the dropdown menu of quality indicators), while they were perceived as not ideal in the interviews, are favorable over the proposed alternatives, due to scalability issues of the latter. Another important finding made during the evaluation was that the indication of interaction capabilities should be as explicit as possible. Thus, I concluded that user tests should be conducted in order to avoid ineffective interactive elements. In addition to these conclusions, I provided a list of lessons learned throughout the implementation. Derived from the evaluation I then provided the strengths and limitations of this work as well as I listed starting points for prospective future research. Therefore, I explicitly noted that the main focus of this work lay on selecting the best design options as well as on giving rationale for them, since only few resources were found that proposed and discussed design options for data quality problems. Ultimately, I encouraged to perform a benchmark on the algorithms as well as to conduct a user study, which would both provide an additional quantitative evaluation of the visual and technical effectiveness of the implemented prototype. Finally, by iteratively designing, implementing, and evaluating proposed visual as well as interactive design options I provided a novel approach for visualizing the identified data quality problems in both, overview and detail.

APPENDIX A

Search Keywords

Table A.1: List of keywords used during litereature research

Round	Search Engine	Keyword
1	Google Scholar	data quality visualization with multivariate time-oriented data
1	Google Scholar	data quality visualization with multivariate temporal data
1	Google Scholar	data quality visualization with multivariate data
1	Google Scholar	data quality visualization with temporal data
1	Google Scholar	data quality visualization with time-oriented data
2	Google Scholar	data quality visualization
2	Microsoft Academic search	data quality visualization
2	Catalog plus	data quality visualization
2	ACM Search	data quality visualization
2	IEEE Explore	data quality visualization
3	Google Scholar	data quality visualization
3	Microsoft Academic search	data quality visualization
3	Catalog plus	data quality visualization
4	Google Scholar	missing value visualizatiion
4	Google Scholar	missing data visualizatiion
4	Microsoft Academic Search	missing data visualization
4	Catalog plus	missing value visualization
4	Google Scholar	extreme values
4	Google Scholar	outlier visualization

4	Microsoft Academic Search	outlier visualization
4	Catalog plus	outlier visualization
4	Google Scholar	erroneous value visualizatiion
4	Google Scholar	invalid value visualizatiion
4	Google Scholar	invalid data visualizatiion
4	Microsoft Academic Search	invalid data visualizatiion
4	Catalog plus	invalid data visualization

List of Figures

List of Tables

Bibliography

[ABG+14] Bilal Alsallakh, Markus Bögl, Theresia Gschwandtner, Silvia Miksch, Bilal Esmael, Arghad Arnaout, Gerhard Thonhauser, and Philipp Zöllner. A visual analytics approach to segmenting and labeling multivariate time series data. *Proc. of EuroVA*, 14:31–35, 2014.

[Agg13] Charu C Aggarwal. An introduction to outlier analysis. In *Outlier Analysis*, pages 1–40. Springer, 2013.

[AKR+10] Elke Achtert, Hans-Peter Kriegel, Lisa Reichert, Erich Schubert, Remigius Wojdanowski, and Arthur Zimek. Visual evaluation of outlier detection models. In Hiroyuki Kitagawa, Yoshiharu Ishikawa, Qing Li, and Chiemi Watanabe, editors, *DASFAA (2)*, volume 5982 of *Lecture Notes in Computer Science*, pages 396–399. Springer, 2010.

[AMST11] Wolfgang Aigner, Silvia Miksch, Heidrun Schumann, and Christian Tominski. *Visualization of time-oriented data.* Springer, 2011.

[Bos12] Mike Bostock. Towards Reusable Charts. `https://bost.ocks.org/mike/chart/`, 2012. (28.07.2014).

[Cho13] Soless Chong. D3: How to show large dataset. `http://stackoverflow.com/questions/18244995/d3-how-to-show-large-dataset`, 2013. (09.08.2016).

[Chr75] Richard E Christ. Review and analysis of color coding research for visual displays. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 17(6):542–570, 1975.

[CWRY06] Qingguang Cui, Matthew O Ward, Elke A Rundensteiner, and Jing Yang. Measuring data abstraction quality in multiresolution visualizations. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):709–716, 2006.

[Dar15] Niclas Darville. Confidence Interval. `https://bl.ocks.org/ndarville/6552457`, 2015. (21.03.2016).

[Dem06] Barry Demchak. Data Quality and Uncertainty Visualization. https://sosa.ucsd.edu/confluence/download/attachments/14745895/Data+Quality.ppt?version=1&modificationDate=1335386460000, 2006. (01.12.2014).

[Dic12] Charles Dickens. *A tale of two cities.* Vintage, 2012.

[Div15] Gangigunta Divya. D3: How to show large dataset. http://stackoverflow.com/questions/32838702/how-to-load-large-amount-of-data-into-a-d3-js-table, 2015. (09.08.2016).

[Eck02] Wayne W Eckerson. Data quality and the bottom line. *TDWI Report, The Data Warehouse Institute*, 2002.

[End10] Craig K Enders. *Applied missing data analysis.* Guilford Press, 2010.

[EPD05] Cyntrica Eaton, Catherine Plaisant, and Terence Drizd. Visualizing missing data: graph interpretation user study. In *IFIP Conference on Human-Computer Interaction*, pages 861–872. Springer, 2005.

[FN07] Paul Führing and Felix Naumann. Emergent data quality annotation and visualization. In *ICIQ*, pages 424–430, 2007.

[For16] Cisco VNI Forecast. Cisco visual networking index: Global mobile data traffic forecast update, 2015–2020 white paper. 2016. (11.08.2016).

[Gem10] Zach Gemignani. Better Know a Visualization: Small Multiples. http://www.juiceanalytics.com/writing/better-know-visualization-small-multiples, 2010. (17.01.2015).

[GGAM12] Theresia Gschwandtner, Johannes Gärtner, Wolfgang Aigner, and Silvia Miksch. A taxonomy of dirty time-oriented data. In *Multidisciplinary Research and Practice for Information Systems*, pages 58–72. Springer, 2012.

[God99] A Blanton Godfrey. *Juran's quality handbook.* McGraw Hill, 1999.

[GS06] Henning Griethe and Heidrun Schumann. The visualization of uncertain data: Methods and problems. In *SimVis*, pages 143–156, 2006.

[HA04] Victoria J Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

[Hua05] Shiping Huang. *Exploratory visualization of data with variable quality.* PhD thesis, Worcester Polytechnic Institute, 2005.

[IBM14] IBM. IBM Pairwise vs. Listwise deletion: What are they and when should I use them? `http://www-01.ibm.com/support/docview.wss?uid=swg21475199`, 2014. (26.01.2015).

[IIC$^+$13] Tobias Isenberg, Petra Isenberg, Jian Chen, Michael Sedlmair, and Torsten Möller. A systematic review on the practice of evaluating visualization. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2818–2827, 2013.

[IP15] Francesca Ieva and Anna Maria Paganoni. Detecting and visualizing outliers in provider profiling via funnel plots and mixed effect models. *Health care management science*, 18(2):166–172, 2015.

[Joh04] Chris Johnson. Top scientific visualization research problems. *IEEE Computer Graphics and Applications*, 24(4):13–17, 2004.

[KK07] Sung-Soo Kim and WJ Krzanowski. Detecting multiple outliers in linear regression using a cluster method combined with graphical visualization. *Computational Statistics*, 22(1):109–119, 2007.

[KPP$^+$12] Sean Kandel, Ravi Parikh, Andreas Paepcke, Joseph M Hellerstein, and Jeffrey Heer. Profiler: Integrated statistical analysis and visualization for data quality assessment. In *Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI)*, pages 547–554. ACM, 2012.

[LBI$^+$11] Heidi Lam, Enrico Bertini, Petra Isenberg, Catherine Plaisant, and Sheelagh Carpendale. Seven guiding scenarios for information visualization evaluation. 2011.

[LCW02] Xiaohui Liu, Gongxian Cheng, and John Xingwang Wu. Analyzing outliers cautiously. *Knowledge and Data Engineering, IEEE Transactions on*, 14(2):432–437, 2002.

[Lor15] Hoa Loranger. Beyond Blue Links: Making Clickable Elements Recognizable. `https://www.nngroup.com/articles/clickable-elements/`, 2015. (05.11.2015).

[LR93] AV Levitin and TC Redman. A model of the data (life) cycles with application to quality. *Information and Software Technology*, 35(4):217–223, 1993.

[MA14] Silvia Miksch and Wolfgang Aigner. A matter of time: Applying a data–users–tasks design triangle to visual analytics of time-oriented data. *Computers & Graphics*, 38:286–290, 2014.

[Mac15] Alan M MacEachren. Visual analytics and uncertainty: Its not about the data. *EuroVis Workshop on Visual Analytics (2015)*, 2015.

[NH06] Matej Novotny and Helwig Hauser. Outlier-preserving focus+ context visualization in parallel coordinates. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):893–900, 2006.

[Nie00] Jakob Nielsen. Is it ok to use arrows to indicate money movement in & out of an account? `https://www.nngroup.com/articles/end-of-web-design/`, 2000. (28.01.2016).

[Oka15] Okavango. Is it ok to use arrows to indicate money movement in & out of an account? `http://bit.ly/2aNTFof`, 2015. (28.05.2015).

[Pan02] Oracle Pandre. Core J2EE Patterns - Data Access Object. `http://www.oracle.com/technetwork/java/dataaccessobject-138824.html`, 2002. (06.04.2016).

[Pan11] Andrew Pandre. What is the Outlier? `http://apandre.files.wordpress.com/2011/08/iqr.jpg`, 2011. (11.04.2016).

[Pan14] Andrei Pandre. Visible Outliers see it 1st. `http://apandre.wordpress.com/visible-data/outliers/`, 2014. (19.01.2015).

[Pen13] Mike Pennisi. Exploring Reusability with D3.js Mike Pennisi. `https://bocoup.com/weblog/reusability-with-d3`, 2013. (27.07.2014).

[PLW02] Leo L Pipino, Yang W Lee, and Richard Y Wang. Data quality assessment. *Communications of the ACM*, 45(4):211–218, 2002.

[Red98] Thomas C Redman. The impact of poor data quality on the typical enterprise. *Communications of the ACM*, 41(2):79–82, 1998.

[RLA+13] Alexander Rind, Tim Lammarsch, Wolfgang Aigner, Bilal Alsallakh, and Silvia Miksch. Timebench: A data model and software library for visual analytics of time-oriented data. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2247–2256, 2013.

[RWX+07] Elke A Rundensteiner, Matthew O Ward, Zaixian Xie, Qingguang Cui, Charudatta V Wad, Di Yang, and Shiping Huang. Xmdvtool q:: quality-aware interactive data exploration. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 1109–1112. ACM, 2007.

[Sad13] Shazia Sadiq. Handbook of data quality. *New York: Springer. do i*, 10:978–3, 2013.

[Saf06] D. Saffer. *Designing for Interaction: Creating Smart Applications and Clever Devices.* Pearson Education, 2006.

[Saf10] Dan Saffer. *Designing for interaction: creating innovative applications and devices.* New Riders, 2010.

[SB92] Manojit Sarkar and Marc H Brown. Graphical fisheye views of graphs. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 83–91. ACM, 1992.

[SCB98] Deborah F Swayne, Dianne Cook, and Andreas Buja. Xgobi: Interactive dynamic data visualization in the x window system. *Journal of Computational and Graphical Statistics*, 7(1):113–130, 1998.

[SEG05] Rajmonda Sulo, Stephen Eick, and Robert Grossman. Davis: a tool for visualizing data quality. *Posters Compendium of InfoVis*, 2005:45–46, 2005.

[Sha05] G Shankaranarayanan. Towards implementing total data quality management in a data warehouse. *Journal of Information Technology Management*, 16(1):21–30, 2005.

[Shn96] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pages 336–343. IEEE, 1996.

[SMM12] Michael Sedlmair, Miriah Meyer, and Tamara Munzner. Design study methodology: Reflections from the trenches and the stacks. *Visualization and Computer Graphics, IEEE Transactions on*, 18(12):2431–2440, 2012.

[SMR98] David M Sebert, Douglas C Montgomery, and Dwayne A Rollier. A clustering algorithm for identifying multiple outliers in linear regression. *Computational statistics & data analysis*, 27(4):461–484, 1998.

[Sti13] Jim Stikeleather. Data Quality and Uncertainty Visualization. `http://blogs.hbr.org/2013/03/when-data-visualization-works-and/`, 2013. (20.09.2014).

[SZ12] G Shankaranarayanan and Bin Zhu. Data quality metadata and decision making. In *System Science (HICSS), 2012 45th Hawaii International Conference on*, pages 1434–1443. IEEE, 2012.

[TA13] Christian Tominski and Wolfgang Aigner. The TimeViz Browser - A Visual Survey of Visualization Techniques for Time-Oriented Data. `http://survey.timeviz.net/`, 2013.

[TAF12] Matthias Templ, Andreas Alfons, and Peter Filzmoser. Exploring incomplete data using visualization techniques. *Advances in Data Analysis and Classification*, 6(1):29–47, 2012.

[TKSK08] Tatiana Tekušová, Martin Knuth, Tobias Schreck, and Jörn Kohlhammer. Data quality visualization for multivariate hierarchic data. *InfoVis demo. http://www. gris. informatik. tu-darmstadt. de/˜ tschreck/papers/infovis08-poster. pdf*, 2008.

[Tuk77] John W Tukey. Exploratory data analysis. 1977.

[UHHS96] Antony Unwin, George Hawkins, Heike Hofmann, and Bernd Siegl. Interactive graphics for data sets with missing values—manet. *Journal of Computational and Graphical Statistics*, 5(2):113–122, 1996.

[Val14] Jim Vallandigham. How to Make Interactive Linked Small Multiples. `https://flowingdata.com/2014/10/15/linked-small-multiples/`, 2014. (15.01.2015).

[vAMPR04] R Hevner von Alan, Salvatore T March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *MIS quarterly*, 28(1):75–105, 2004.

[VdVK04] DE Van de Vlag and MJ Kraak. Multivariate visualization of data quality elements for coastal zone monitoring. In *ISPRS Conference Proceedings Commission IV, ISPRS*, pages 15–23, 2004.

[WS96] Richard Y Wang and Diane M Strong. Beyond accuracy: What data quality means to data consumers. *J. of Management Information Systems*, 12(4):5–33, 1996.

[WW07] Shouhong Wang and Hai Wang. Mining data quality in completeness. In *ICIQ*, pages 295–300, 2007.

[WXYR11] Matthew Ward, Zaixian Xie, Di Yang, and Elke Rundensteiner. Quality-aware visual data analysis. *Computational Statistics*, 26(4):567–584, 2011.

[XHWR06] Zaixian Xie, Shiping Huang, Matthew O Ward, and Elke A Rundensteiner. Exploratory visualization of multivariate data with variable quality. In *2006 IEEE Symposium On Visual Analytics Science And Technology*, pages 183–190. IEEE, 2006.

[Yau10] Nathan Yau. 11 Ways to Visualize Changes Over Time – A Guide. `http://goo.gl/vHhX9L`, 2010. (21.01.2015).

[Zha14] Boxun Zhang. Does D3.js work efficiently on massive amounts of data (200 million rows with at least a dozen columns)? `http://bit.ly/2babDQY`, 2014. (09.08.2016).

[ZSC07] Bin Zhu, G Shankar, and Yu Cai. Integrating data quality data into decision-making process: an information visualization approach. In *Human Interface and the Management of Information. Methods, Techniques and Tools in Information Design*, pages 366–369. Springer, 2007.

[ZWRS07] Xie Zaixian, Matthew O Ward, Elke A Rundensteiner, and Huang Shiping. Integrating data and quality space interactions in exploratory visualizations. In *Coordinated and Multiple Views in Exploratory Visualization, 2007. CMV'07. Fifth International Conference on*, pages 47–60. IEEE, 2007.