Visplause: Visual Data Quality Assessment of Many Time Series Using Plausibility Checks

Clemens Arbesser, Florian Spechtenhauser, Thomas Mühlbacher, and Harald Piringer

Abstract—Trends like decentralized energy production lead to an exploding number of time series from sensors and other sources that need to be assessed regarding their data quality (DQ). While the identification of DQ problems for such routinely collected data is typically based on existing automated plausibility checks, an efficient inspection and validation of check results for hundreds or thousands of time series is challenging. The main contribution of this paper is the validated design of Visplause, a system to support an efficient inspection of DQ problems for many time series. The key idea of Visplause is to utilize meta-information concerning the semantics of both the time series and the plausibility checks for structuring and summarizing results of DQ checks in a flexible way. Linked views enable users to inspect anomalies in detail and to generate hypotheses about possible causes. The design of Visplause was guided by goals derived from a comprehensive task analysis with domain experts in the energy sector. We reflect on the design process by discussing design decisions at four stages and we identify lessons learned. We also report feedback from domain experts after using Visplause for a period of one month. This feedback suggests significant efficiency gains for DQ assessment, increased confidence in the DQ, and the applicability of Visplause to summarize indicators also outside the context of DQ.

Index Terms—Data Quality Assessment, High-Dimensional Data, Hierarchical Aggregation, Linked Views

1 Introduction

Ensuring a sufficient data quality (DQ) is essential for data-driven tasks like analysis, modeling, and reporting. Problems such as missing data, wrong data, and duplicates may prevent an application of analytical methods or may cause unusable or misleading results [12]. DQ management is thus of central practical relevance. However, numerous sources emphasize that ensuring an appropriate DQ is time-consuming [18] and requires the involvement of domain experts [13]. DQ may account for up to 80% of the time and cost in data warehousing projects [5].

The background and initial motivation of this work are challenges regarding DQ assessment in the energy sector. In this application domain, measurements of power generation, consumption, and meteorological quantities are continuously acquired. The DQ of the corresponding time series needs to be assessed before subsequent processing steps like forecasting and accounting. The identification of DQ problems for such routinely collected data is typically based on automated plausibility checks ranging from simple validation routines to advanced approaches for anomaly detection [4, 13]. In case of a small number of checks for a few sensors, the inspection of check results using standard visualizations such as time series plots is feasible.

In recent years, however, the number of sensors and corresponding time series has been increasing rapidly due to trends like decentralized energy production. For example, transmission system operators need to process data from hundreds or even thousands of small-scale power plants such as photovoltaic plants, windmills, and small hydropower plants in regular intervals, e.g., weekly or monthly. With the advent of smart meters, the number of sensors will further explode by multiple orders of magnitudes. Similar explosions in the dimensionality of acquired sensor data can also be observed in other application contexts such as advanced manufacturing. This suggests that the problem of DQ assessment in many time series is of general importance. Besides the sheer number of involved time series, additional challenges of DQ assessment include the diversity of plausibility checks, the validation of automated anomaly detection results, and the usually very tight time constraints of operators and analysts for a detailed inspection of the data. While communicating the results of analytical techniques for assessing DQ is considered an important research topic in general [18], these particular challenges have not yet been sufficiently addressed by the visualization literature so far.

In this paper, we argue that meta-information about the time series and plausibility checks can significantly contribute to an effective inspection of DQ problems for large numbers of time series. For power generation time series, for example, meta-information includes the location of the power-plant as well as its type, operator, and so on. Meta-information also refers to the plausibility checks themselves and involves, e.g., the type of detected DQ problem and check-specific parameters. This meta-information enables to structure the potentially large number of plausibility checks in a way that is both flexible and meaningful to the analyst.

The **main contribution** of this paper is the validated design of *Visplause*, a system to support an effective inspection of DQ problems in many time series. The key idea of Visplause is to utilize meta-information about time series and plausibility checks for structuring and summarizing results of DQ checks in a flexible way. Linked views enable users to inspect anomalies in detail and to generate hypotheses about possible causes.

Following guidelines in design study methodology [33], additional contributions include:

- A characterization of involved tasks and a set of identified goals that guided the design of Visplause.
- Detailed reflections on the design process by explicitly discussing design decisions and identifying lessons learned at four design stages.
- Qualitative feedback from domain experts in the energy sector after using Visplause for a period of one month.

2 Tasks and Data

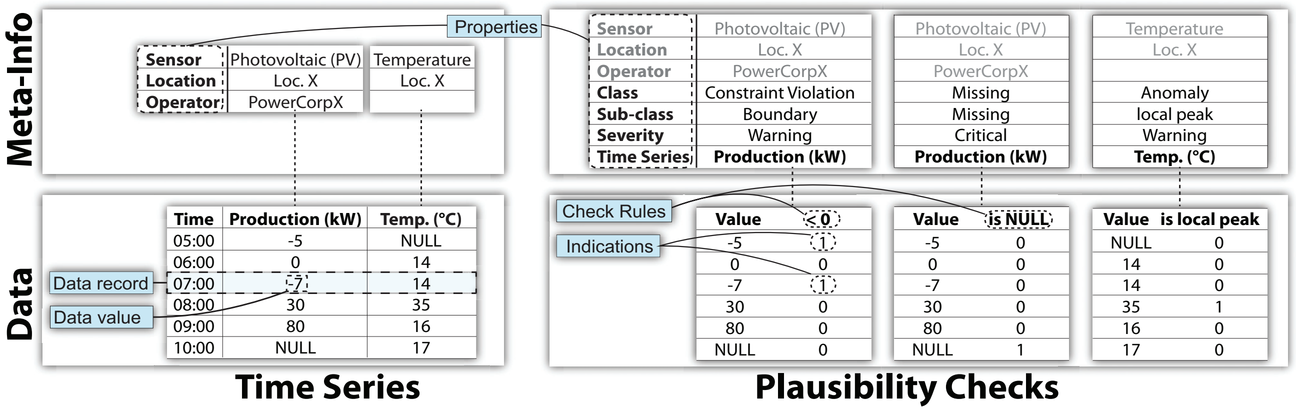
We started this project with a thorough analysis of the work domain. Based on collaborations with two partner companies in the energy sector, we identified recurring DQ assessment tasks (Sec. 2.1), from which we derived our design goals (Sec. 2.2). Afterwards, we provide a description of the data model assumed in this paper (Sec. 2.3).

2.1 Task Analysis

This work is motivated by an ongoing collaboration with partners in the energy sector since 2012, i.e., a transmission system operator and an IT-solution provider servicing more than 40 energy companies. Domain experts in these companies utilize regularly acquired time series data for power grid control and risk management. Most of their tasks

- *The authors are with the VrVis Research Center, Donau-City-Str. 1, 1220 Vienna, Austria. Email: {arbesser, fspechten, tm, hp}@vrvis.at*

Manuscript received 31 Mar. 2016; accepted 1 Aug. 2016. Date of publication 15 Aug. 2016; date of current version 23 Oct. 2016.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TVCG.2016.2598592



Sensor	Photovoltaic (PV)	Temperature
Location	Loc. X	Loc. X
Operator	PowerCorpX	

Time	Production (kW)	Temp. (°C)
05:00	-5	NULL
06:00	0	14
07:00	-7	14
08:00	30	35
09:00	80	16
10:00	NULL	17

Sensor	Photovoltaic (PV)	Photovoltaic (PV)	Temperature
Location	Loc. X	Loc. X	Loc. X
Operator	PowerCorpX	PowerCorpX	
Class	Constraint Violation	Missing	Anomaly
Sub-class	Boundary	Missing	local peak
Severity	Warning	Critical	Warning
Time Series	Production (kW)	Production (kW)	Temp. (°C)

Value	< 0
-5	1
0	0
-7	1
30	0
80	0
NULL	0

Value	is NULL
-5	0
0	0
-7	0
30	0
80	0
NULL	1

Value	is local peak
NULL	0
14	0
14	0
35	1
16	0
17	0

Fig. 1. The data model of Visplause. Time series and plausibility checks carry meta-information. Plausibility checks mark incoming time series as 'check indication' or 'no indication' based on the automated evaluation of check rules.

involve a daily monitoring and DQ assessment of newly acquired data (e.g., prices, consumption, weather, etc.). Data-driven modeling activities such as building or validating forecast models based on longer time series are typically performed in less frequent intervals (e.g., monthly). The routine DQ assessment of newly acquired data based on plausibility checks is an important, frequent, and time-critical activity.

Throughout four years of collaboration, insights gained from joint data analysis sessions, semi-structured interviews, and contextual inquiries [15] with 11 domain experts from multiple business units enabled us to identify recurring tasks of DQ assessment. While DQ assessment is a multi-faceted topic and previous work has addressed certain subtasks (e.g., [11, 17]), we stress that our focus are tasks concerning the *regular* inspection and validation of identified DQ problems using *existing* plausibility checks. The detection of previously unidentified problems, as well as the correction of problems by modifying or imputing data, are out of scope.

T1 - Routine DQ assessment for a specific use
This task is usually done as preparation for downstream processing (see T2 and T3) or for reporting. In some cases, the DQ may be insufficient for certain uses and sufficient for others. This task entails the following sub-tasks, with concrete examples in italics:

- **Summarization** - Assess the overall DQ in terms of the proportion of time series and data records afflicted by DQ problems. *"Can the data be used as it is, or are there DQ issues to investigate further before proceeding with downstream tasks?"*
- **Drill-down** - Investigative drill-down on DQ problems by time series or checks to (1) localize problems and to (2) compare problem occurrence across different sources. *"Which time series are afflicted by the issues indicated by the summary? Is time series X more afflicted than time series Y?"*
- **Temporal characterization** - Assessment of the temporal distribution of check indications concerning linear aspects (e.g., gradually increasing problem frequencies) and cyclic aspects (e.g., peaks at certain times of day). *"Is this an isolated anomaly, or does it occur in a regular pattern?"*

T2 - Hypothesis generation about DQ problem causes
Causes of problems may refer to the data-generating process (i.e., the data source), the data transformation and management, or to the plausibility checks themselves in case of false positives. The task may entail further investigation of the hypotheses and the communication to responsive stakeholders (e.g., data providers, IT departments).

- **Inspection of details** - Inspect specific time series and single values for investigating and validating DQ problems. *"This pattern looks like a gradually failing sensor."*, or *"These are all false positives. Maybe we need to recalibrate the check?"*
- **Correlation analysis** - Identify potential relationships between indication patterns and possible explanatory time series. *"Are there any weather conditions in which this problem occurs particularly often?"*

T3 - DQ-aware selection and export of data
This task involves an efficient selection and export of subsets of data records and time series that satisfy certain DQ constraints for downstream processing or upstream reporting, without introducing a selection bias. *"For training a forecast model, time series with too many outliers can not be used. Is my selection of clean data records still representative for all weather conditions/months of the year?"*

2.2 Design Goals

The following set of goals guided the design process of our proposed system. Most aspects of these goals were established based on the task analysis, while some aspects became clearer throughout the design process, as described in Section 5.

G1 - Fast and efficient overall DQ assessment
To support the efficient assessment of routinely collected data (T1), the system should efficiently summarize the overall DQ. This entails appropriate default configurations and support for sorting and filtering.

G2 - Flexible in-depth analysis of check results
The identified tasks (T1-T3) involve an assessment of the DQ on multiple aggregation levels: (1) *Global* summarization (e.g., overall DQ), (2) *Local* summarization in terms of meaningful subsets of time series (e.g., DQ per power plant), types of checks (e.g., DQ per problem type) and temporal aspects (e.g., DQ per month), and (3) on the level of *individual* DQ issues (e.g., outliers). The choice of and the navigation between aggregation levels should be flexible, fast and intuitive. The system should also support the selection of data subsets on any of the available aggregation levels (T3).

G3 - Scalability to large numbers of time series, checks and individual observations
In the energy sector, data is often acquired by a large number of sensors (> 100) over long periods of time (up to several years) and/or at high frequencies (e.g., one measurement per second). Therefore, the system design should scale for up to hundreds of time series, each with up to millions of data values and multiple plausibility checks.

G4 - Scalable visual complexity
To accommodate tasks of varying complexity and users of varying experience, the visual complexity of the system needs to be adjustable. Regular monitoring and reporting tasks should be easy to accomplish, while expert users should not be limited regarding view configuration.

G5 - Workflow integration
The system should naturally fit into the DQ management workflow of analysts in the energy sector. It should therefore support (1) the import of time series with meta-information via common exchange formats like CSV or SQL, (2) the re-use of existing plausibility checks, and (3) the export of currently selected data (T3). In favor of a broad applicability, the system should allow making use of existing semantic information about time series or checks, but it should not require the presence of any *particular* meta-information property.

2.3 Data Model

Fig. 1 illustrates a very simple example of a typical data model in the context of energy-related time series. The model distinguishes time series and plausibility checks on these time series. In related data quality literature, plausibility checks are often termed 'quality checks' or 'anomaly detection algorithms' [11, 17]. Each plausibility check (subsequently abbreviated as **check**) evaluates a rule or anomaly detection algorithm to test values from one or more time series for the presence of a particular DQ problem. The result of each check is a binary classification of data values as 'indication' or 'no indication'.

Time series and checks have meta-information, consisting of **properties** such as the sensor type (distinguishing power production from meteorological quantities), the location, the operator, and many more. Properties may also be undefined for some time series. For example, meteorological sensors do not have a power plant operator. Checks adopt non-ambiguous meta-information of their underlying time series and have additional properties such as the severity or classification of the detected DQ problem. This enables an application-specific incorporation of various DQ taxonomies (see Sec. 3).

We emphasize that the meta-information properties refer to time series and checks themselves, not to individual data values. Similar to work by Turkay et al. [45], meta-information represents an orthogonal aspect which is necessary to structure the potentially very high-dimensional data table of the actual time series and checks. We also note that a very similar distinction of energy time series vs. meta-information has been made in recent work by Brehmer et al. [3], however, their data model does not include plausibility checks.

3 Related Work

Data quality (DQ) has long been a topic of intensive research in multiple fields [32]. A widely agreed definition of the term 'data quality' from the consumer's point of view is *fitness for use* [46].

There are different approaches to characterizing DQ. Multiple researchers identified dimensions of DQ [2, 28] such as availability, consistency, and reliability. Other taxonomies focus on the classification of particular problems like missing, wrong, and unusable data [22] and their sources [30]. Gschwandtner et al. [12] summarize several taxonomies and provide an own taxonomy of dirty time-oriented data. A classification of DQ issues according to most of these taxonomies can be represented by our data model using check meta-information properties (see Sec. 2.3).

The automated detection of DQ problems such as anomalies has long been a key topic in statistics and data base management [4, 13]. Identifying possible DQ problems, however, often requires an intuition of the data and possible problems. Exploratory visualization is suitable to provide this intuition [20, 48]. In this sense, well-known systems like Tableau [40], Spotfire [44], GGobi [39], and many others are all relevant for DQ assessment. While these systems provide versatile general-purpose aggregation of data dimensions, we found that the support for structuring high-dimensional data was limited (G3).

Kandel et al. [18] describe DQ assessment as a visualization opportunity in the broader context of data wrangling. TimeCleanser [11], for example, supports the process of correcting DQ issues in time-oriented data, but offers limited flexibility for analyzing the results of plausibility checks. Most importantly, our focus is on the inspection of check results, not on the modification of data.

Multiple approaches explicitly convey DQ aspects during visual analysis. Ward et al. [47] describe strategies for the measurement, display, and utilization of quality aspects at all stages of the visualization pipeline. Sulo et al. [38] highlight DQ problems like duplicates and missing values in a tabular visualization on a data-record level, but do not support aggregation, which limits the scalability. Other work focuses on missing data. This issue is repeatedly mentioned as important for visual analytics systems [18, 49] and there are several studies on displaying missing data [7, 9, 42, 43]. However, most of this work adds information to existing visualizations and does not provide overview and drill-down (G2) regarding data completeness.

Several approaches have been suggested to visualize data anomalies in sensor networks. Shi et al. [35] facilitate sensor failure diagnosis by expert users based on topological, correlational, and dimensional views of anomalies. Steiger et al. [37] use dimension reduction for identifying anomalous patterns. Both focus on identifying anomalous patterns, whereas we assume the existence of suitable plausibility checks and focus on the analysis of their results (G2). In this paper, we adopt the suggestion of Turkay et al. [45] to use structural meta-information about high-dimensional datasets in order to break down the data dimensions into a manageable number of meaningful subsets.

Several commercial systems provide coarse summaries of DQ indicators also for non-expert users. The Talend Open Studio for Data Quality [41] visualizes the results of pre-defined or user-defined quality indicators and highlights DQ problems in corresponding tables. IBM Watson Analytics [16] assigns an overall DQ score to a data set and to individual data attributes. However, both Talend and IBM Watson rely on mostly static diagrams for result presentation with limited possibilities for drill-down or an interactive validation of plausibility check results.

Many visualization systems support anomaly detection. Janetzko et al. [17], for example, visualize the results of an anomaly detection of power consumption data. Their pixel-based approach uses topological information about the data sources to draw a quadtree. We regard Profiler [19] as the most related system. Profiler uses data mining methods to automatically flag problematic data for a visual summary, structured by problem class. Mutual information measures suggest binned views for a detailed yet scalable analysis of these problems. However, neither of these systems support a flexible DQ analysis on multiple aggregation levels using more general meta-information of both data attributes and checks (G2). Moreover, the schema and anomaly browsers of Profiler only provide a list of data attributes, which limits its applicability for the DQ assessment of many time series.

4 Description of Visplause

This section describes our system Visplause based on a guiding data example from the energy sector. The intention is to illustrate the applicability of the system to address the identified tasks, while Sec. 5 provides detailed design justifications with respect to alternatives considered in the design process.

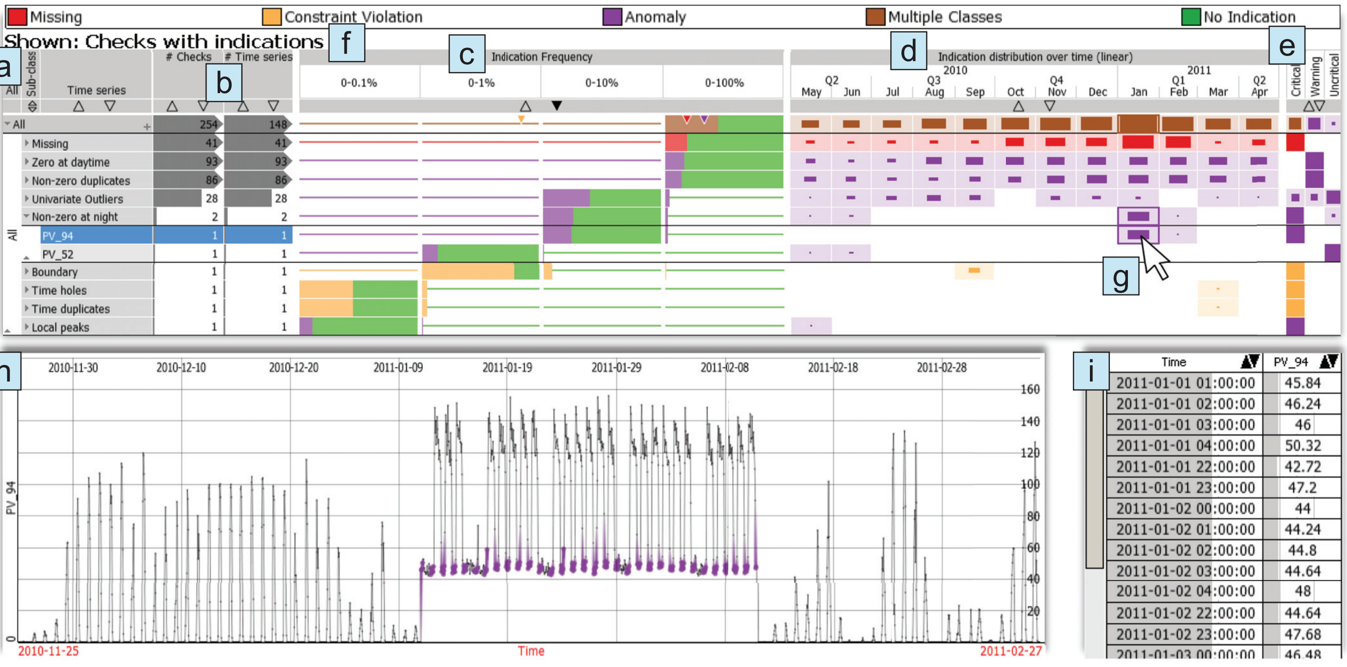
4.1 Guiding Example: Photovoltaic Power Plants

As a guiding example throughout this paper, we use a real but anonymized data set from the energy sector. For approximately one year, the data comprises the hourly production of 95 photovoltaic power plants (time series PV01 - PV95), as well as hourly meteorological measurements such as global radiation or temperature from multiple weather stations. In total, the data consists of 160 time series and approximately 9000 data records. As meta-information, all time series carry the type of the sensor they originated from as a property 'Sensor', as well as a property 'Location'.

This data is a typical example of sensor measurements that require regular DQ assessment prior to tasks such as statistical modeling. Monitoring tasks often involve shorter time periods, e.g. a few consecutive days, or a week from this one-year example. In cooperation with domain experts, we have identified a set of DQ problems that should be tested for this data using plausibility checks. We do not argue for a general validity of this classification.

- **Missing**: data values being NULL
- **Anomaly** (data values that are not *impossible* but *improbable* given the data context)
  - *Zero at daytime*: power production being zero at daytime
  - *Non-zero at night*: non-zero power production at night
  - *Univariate outliers*: meteorological data values which are outliers with respect to their univariate normal distribution
  - *Non-zero duplicates*: repeating identical sensor data values other than zero (indication for malfunctioning sensor)
- **Constraint Violation** (semantically impossible data values or relations between values)
  - *Boundary*: e.g., negative values for power production
  - *Time holes*: time steps of more than one hour
  - *Time duplicates*: duplicate time stamps

In total, 529 plausibility checks were defined by applying this set of rules to the time series where applicable. Not all DQ issues are equally important. In this data set, our domain experts distinguish three levels



All	Sub-class	Time series	# Checks	# Time series
All			254	148
All	Missing		41	41
All	Zero at daytime		93	93
All	Non-zero duplicates		86	86
All	Univariate Outliers		28	28
All	Non-zero at night		2	2
All		PV_94	1	1
All		PV_52	1	1
All	Boundary		1	1
All	Time holes		1	1
All	Time duplicates		1	1
All	Local peaks		1	1

Time	PV_94
2011-01-01 01:00:00	45.84
2011-01-01 02:00:00	46.24
2011-01-01 03:00:00	46
2011-01-01 04:00:00	50.32
2011-01-01 22:00:00	42.72
2011-01-01 23:00:00	47.2
2011-01-02 00:00:00	44
2011-01-02 01:00:00	44.24
2011-01-02 02:00:00	44.8
2011-01-02 03:00:00	44.64
2011-01-02 04:00:00	48
2011-01-02 22:00:00	44.64
2011-01-02 23:00:00	47.68
2011-01-03 00:00:00	46.48

Fig. 2. Visplause consists of multiple linked parts. The DQ overview provides a hierarchical overview of check results (a). In this example, checks are aggregated by 'Sub-class' and 'Time Series' to reveal the most prevalent types of DQ issues and their source. DQ overview columns show additional information such as (b) the number of checks and time series, (c) the percentage of affected data, (d) the distribution of check indications over time and (e) the severity of check indications. (f) The current filter shows only checks with at least one check indication. (g) *Non-zero at night* anomalies at power plant PV_94 in January 2011 have been selected. (h, i) Linked views provide details of the selected data anomalies for validation.

of problem severity: 'uncritical', 'warning' and 'critical'. All 'Missing' checks are considered 'critical'. The rules of most 'Anomaly' problems are evaluated for different thresholds by one check per severity level. For example, 'Non-zero at night' considers values only marginally above zero as 'uncritical', but larger values as 'warning' and eventually 'critical'.

To incorporate these semantics as meta-information, all checks have a property 'Severity' reflecting the severity level, and properties '**Class**', and '*Sub-class*' reflecting the first and second level of the classification above. Additionally, checks inherit the properties of the time series they are referring to (see Fig. 1).

4.2 System Overview

The system Visplause consists of multiple linked parts. The core element is the **Data Quality Overview**, an interactive visualization for the inspection of plausibility check results (Fig. 2a-g). The key idea is to visualize the checks using a hierarchical tabular layout. Each table row shows the aggregated check results for a particular semantic group of checks, e.g., checks detecting the same problem class, or checks based on the same underlying time series. Grouping checks in a nested way by such meta-information properties defines a scalable hierarchy on the checks (G3) which is shown using an indented layout at the left-hand side of the view (Fig. 2a).

Additional columns to the right can be shown or hidden as needed (G4). These columns visualize information per row such as the number of aggregated checks or time series (Fig. 2b), the frequency of indications (Fig. 2c), the temporal distribution of indications (Fig. 2d), or the severity level of indications (Fig. 2e). Sorting the table by indication frequency guides the user to check groups that are particularly afflicted by indications. To focus the overview on relevant parts, checks without any indications are filtered from the list per default (Fig. 2f). Linked views enable the user to inspect check indications in detail (Fig. 2h,i). They include a time series view, a spreadsheet view, and a statistical summary view. In particular, these views provide insights about the correctness and completeness of the checks and may help to find causes of DQ problems.

A typical workflow (G5) starts with the import of data and the evaluation of a pre-defined set of plausibility checks for visualization in a pre-configured set of views (e.g., see Fig. 2). Users may also flexibly parametrize and layout views at any time as described below.

4.3 Task-oriented Description

This section illustrates how Visplause can be used by domain experts to address the tasks described in Sec. 2. While the described usage scenarios are not transcribed from actual analysis sessions, they *do* represent realistic workflows according to our domain experts.

4.3.1 T1: Routine DQ assessment for a specific use

Sarah is a data analyst in the energy sector. She has just received the yearly photovoltaic power plant data from the data providers and wants to assess the data fitness for clustering the power plants. She imports the data and a predefined set of plausibility checks (see Sec. 4.1).

Initially, Visplause shows an intentionally simple overview of all check results (Fig. 3a). The DQ Overview supports the fast assessment of overall data fitness (G1) with the **Indication Frequency Column**. This column shows the percentage of data records afflicted by indications. Stacked bars encode the proportion of records with and without indications by checks in that row.

Sarah discovers that almost half of all data records have indications by one or more of the 254 underlying checks (Fig. 3a). Tooltips provide the exact indication frequency on demand (44.8%). She wants to investigate which time series are affected. The DQ Overview offers to dynamically specify a check hierarchy to support this task. By clicking on the '+' sign next to the root node 'All', Sarah can perform a drill-down by different check properties, allowing her to flexibly summarize check results (G2). Her drill-down by 'Time Series' reveals that eight photovoltaic power plants (abbr. PVs) have anomalies or missing values in more than 5% of their data (Fig. 3b). Scale-stack bars [14] ensure that even small amounts of indications are clearly visible, as the smallest order of magnitude containing the relative indication frequency value displays the stacked bar in full height.

The color of indication frequency bars reveals that indications of these eight PVs are either caused by anomalies (purple) or by more than one check class (brown). In the latter case, small triangles on top of the bars show the frequencies of involved check classes in the respective magnitude. To determine which checks have indications for a particular time series (PV_03), Sarah performs a local drill-down

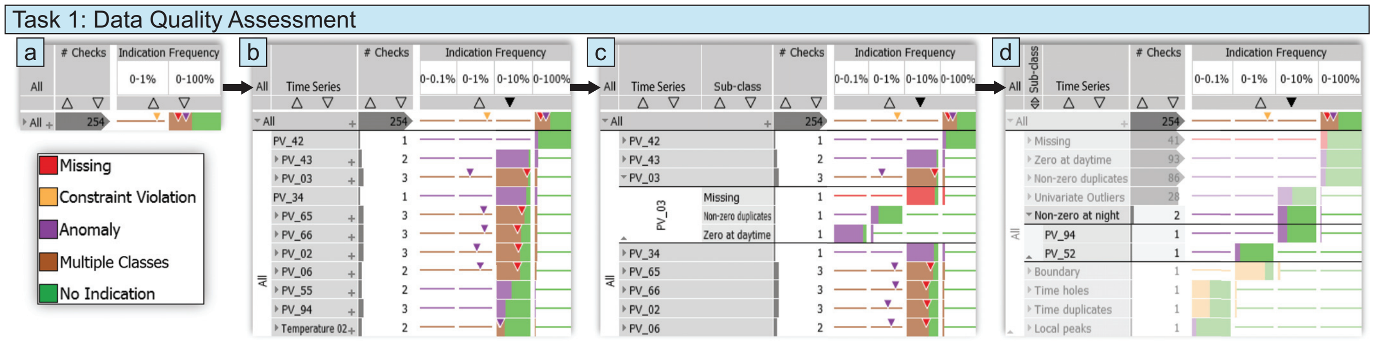


Fig. 3. The frequency of check indications at four stages of a drill-down scenario. (a) Aggregation of all checks, (b) drill-down by time series, (c) a local drill-down on the attribute 'PV_03' by check classification, and (d) another local drill-down after swapping the hierarchy levels.

by 'Sub-class' (Fig. 3c) by clicking on the '+' sign next to the node 'PV_03'. Its check indications mainly comprise missing values ($\approx$9%) and only a few data anomalies ($< 1\%$). Sarah wonders if this distribution of check indications across sub-classes is generally true. She swaps the hierarchy levels by drag and drop such that drill-downs are performed first by 'Sub-class' and then by 'Time Series' (Fig. 3d). This confirms that missing values are the most frequent check indication overall, followed by 'zero at daytime' and 'non-zero duplicate' anomalies, which occur similarly often.

For most subsequent tasks, time series data should not contain long stretches of anomalous data. Visplause supports the visualization of the temporal distribution of check indications with the temporal **Indication Distribution Column**. This column visualizes the percentage of data records with check indications in semantically meaningful temporal partitions by the size of centered rectangles (Fig. 2d). Centering the rectangles aligns them both horizontally and vertically, thus supporting comparisons between adjacent rows and columns. The partitioning is either linear (e.g., one partition per month) or cyclic (e.g., one partition per day of week), with a user-defined degree of detail.

Sarah adds a temporal Indication Distribution Column to the DQ overview (Fig. 2d). In most time series, indications occur throughout the year, with a noticeable increase in winter 2010 and spring 2011. However, the check 'Non-zero at night' shows a very noticeable peak of check indications in January 2011 (Fig. 2g).

How critical are these check indications? Sarah adds the **Severity Level Column** to find out. Similar to Indication Distribution Columns, such check property columns use the size of centered rectangles to visualize the relative frequency of check indications per category, in this case per 'Severity' level. Sarah sees that all 'Non-zero at night' indications of PV_94 are *critical*. She decides to inspect the data to validate this finding. To this end, she selects all check indications of PV_94 in January 2011 with a click (Fig. 2g). As a result, a linked Time Series view automatically shows the underlying time series PV_94 and highlights the selected indications (Fig. 2h), while the spreadsheet view shows the exact values (Fig. 2i). This reveals that these indications are caused by a significant offset for a long period of time. Sarah decides to report this finding to her data providers. Concluding T1, the DQ seems sufficient overall, but some parts of the data must be excluded for certain downstream tasks.

4.3.2 T2: Hypothesis generation about DQ problem causes

Previously, Sarah discovered that missing values are the most frequent check indication overall (Sec. 4.3.1). She is particularly interested in missing power production values. What caused these DQ issues? Can other time series provide further insights?

To obtain an overview of missing values for each power production time series, Sarah defines the hierarchy based on problem class, sensor type, and time series (Fig. 4a). As all checks without indications are filtered from the DQ Overview per default, the resulting hierarchy shows that five of the 95 PVs have missing values (Fig. 4a).

Sarah hypothesizes that certain meteorological conditions may cause sensor malfunction and thus missing sensor data. Visplause provides the **Statistical Summary View** to investigate hypotheses about possible causes of DQ issues (Fig. 4b). This view is an adapted ver-

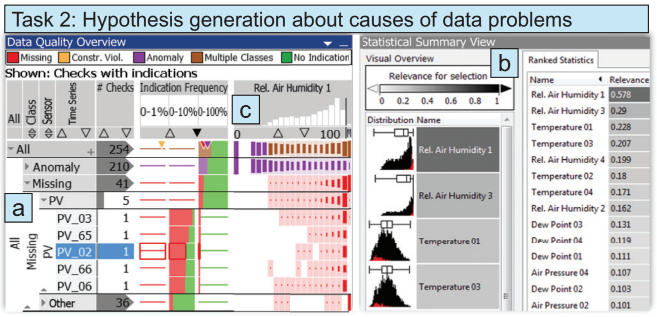


Fig. 4. Visplause as used for hypothesis generation regarding missing data of power plant production (a). Five power plants have missing values. Ranking meteorological quantities by their relevance for the selected missing values of PV_02 suggests air humidity as possible cause (b). Adding air humidity as column to the DQ Overview (c) shows a relationship between high air humidity and missing values of power plants.

sion of the 1D rank-by-feature framework [34]. The key concept is to show histograms and statistics for each of a potentially large number of time series, which may be ranked by any of these statistics. Specifically, the view provides the *information gain* as a statistical measure to quantify the mutual information between each time series and an ad-hoc classification of data records given by a user-defined selection of indications (see Kandel et al. [19] for a similar concept). Ranking by this relevance measure guides the user towards possible explanatory time series, as the measure is high for time series where selected data records are distributed very differently from non-selected data records.

Sarah selects the missing values of PV_02 in the DQ Overview by clicking the respective table row. This causes the Statistical Summary View to compute the relevance of time series for this selection. Sarah discovers that relative air humidity of weather station 1 ranks first (Fig. 4b). She decides to investigate the relationship between this humidity time series and missing PV values in general.

For visualizing relationships between check indications and values of time series, the DQ Overview offers quantitative and categorical **Indication Distribution Columns**. Quantitative distribution columns partition the value range of time series (e.g., humidity values) in equally sized steps with intuitive boundaries (e.g., from 0 to 100 in steps of 5). Categorical time series create one partition per category or per user-defined subset of categories (e.g., working days vs. weekends). Similar to other distribution columns, the relative frequency of check indications in each partition is visualized by scaling a centered rectangle. A histogram in the column header shows the number of data records per partition. 'M' denotes a partition for missing values.

Sarah adds a quantitative Distribution Column for air humidity of weather station 1 to the DQ overview (Fig. 4c). The relation between high humidity values and missing values of power plants, particularly PVs 2, 3, and 6, is immediately apparent. She concludes that these check indications might be caused by hardware failures at high air humidity and reports this finding to the respective power plant operators.

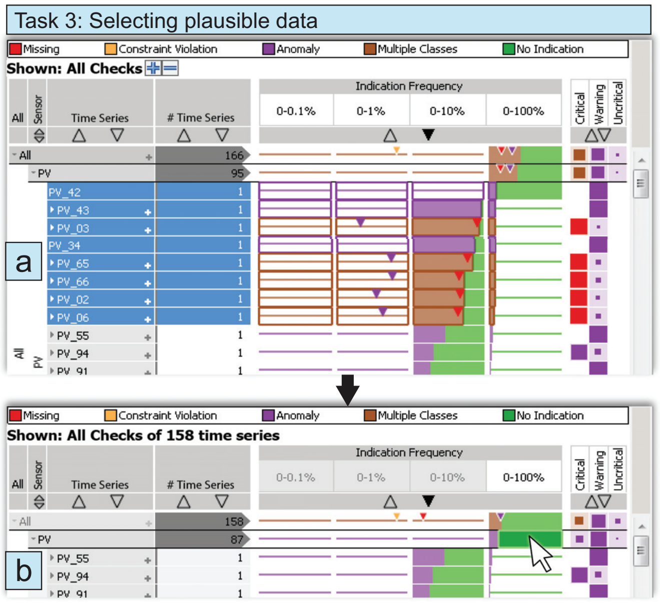


Fig. 5. Using the DQ Overview to select clean data for further processing. Initially, the view reveals eight power plants with > 5% check indications and a severity level of warning or critical (a). After excluding these power plants via filtering, the clean data of the remaining power plants can be selected with a single click (b).

4.3.3 T3: DQ-aware selection and export of data

Sarah wants to cluster photovoltaic power plants in an external tool by their similarity during normal operation (i.e., only plausible values). She thus needs to exclude data records with indications, as well as irrelevant time series (e.g., meteorological measurements).

As a first step, Sarah wants to focus the DQ Overview on time series referring to photovoltaic power plants. She thus defines the hierarchy based on the sensor type (e.g., 'PV', 'Temperature', etc.) and the underlying time series (Fig. 5a).

Next, Sarah wants to exclude power plants with too many DQ problems entirely. After sorting by indication frequency, the DQ Overview shows that eight power plants have indications in more than 5% of their data. The Severity Level Column reveals that these indications are all classified as either 'warning' or 'critical'. Sarah decides to exclude these power plants for subsequent processing steps. Removing them from the DQ Overview is supported by interactive filtering of checks. To do so, Sarah first selects their rows, which highlights them in blue. She then clicks the "-" sign of the filter on top of the DQ overview and selects to remove all checks referring to any of the selected time series from the view. As a result, the DQ Overview now only shows PV-time series with less than 5% check indications (Fig. 5b).

In order to select the clean data records of all remaining PVs for clustering, Sarah clicks on the green bar in the Indication Frequency Column of the aggregated row 'PV' (Fig. 5b). The resulting clean data matrix can be inspected in the Spreadsheet View and exported, e.g., via the clipboard to other tools (G5). In use-cases where avoiding a selection bias is crucial, Sarah could have considered Indication Distribution Columns or linked views before exporting the data. As described in Sec. 4.3.2, the information gain ranking of the Statistical Summary View would guide her towards time series where her selection of clean data does not cover the entire value range.

5 Reflections on Design Evolution

This section describes and reflects upon the design process of Visplause. The goal is to enable a longitudinal comparison of the evolving design along four milestone stages, and to justify design decisions with respect to alternatives that were considered but ultimately discarded in the course of the project. As a result of reflection, we state lessons learned from own mistakes while designing Visplause.

5.1 Stages of the Design Process

The design process of Visplause started in late 2013, and can be characterized as a sequence of four milestone stages:

Initial Design: After an initial task analysis (Sec. 2.1), we used parallel prototyping [6] of hand-drawn sketches to discuss early design decisions with domain experts of our two partner companies. As one example, Fig. 6 shows an initial (and incomplete) sketch of the DQ Overview – the key element of Visplause. During a session of $\sim$ 3 hours, 11 domain experts from the two companies were prompted to give positive, negative, and prospective feedback based on our sketches using the rose-bud-thorn method [24].

Early Prototype: The first prototype of Visplause was implemented based on feedback on the sketches. Its application to real data refined our understanding of tasks and requirements, and exposed deficiencies of previous visual encoding choices. The prototype depicted in Fig. 7a already accounts for some of these insights. In addition to collecting feedback from 6 experts of our partner companies, we showcased the prototype among other projects at 'E-World 2015', Europe's premier energy fair ($\approx$ 24,000 visitors). Around 50 experts from 12 companies including managers and technical directors gave informal feedback after private, grouped demonstrations of 20-30 minutes.

Intermediate Design: Based on insights from the previous stage, we refined the prototype over a period of several weeks. Fig. 7b shows a design iteration that is already very close to the final design (with the exception of the Indication Frequency Column). The evaluation of the design was again based on qualitative feedback collected after interactive sessions with 6 experts from our partner companies.

Final Design: Fig. 2 shows the final design of the DQ Overview, as described in Section 4. It was evaluated based on qualitative feedback collected after a deployment of Visplause from five domain experts from our two partner companies (see Sec.7).

5.2 Design Decisions and Lessons Learned

This section provides detailed justifications for ten major design decisions made in the course of the project. We also identify ***lessons learned (LL)*** from own mistakes and insights while designing Visplause. While we believe that some aspects might be valid in general, we emphasize that this discussion is solely based on a subjective reflection of the design process of Visplause with our domain experts.

Tabular layout of the DQ Overview: The key idea of providing an overview of check results using a table-based layout was already envisioned in the first hand-drawn sketches of Visplause (Fig. 6). We chose table-oriented displays because (1) our users were already familiar with them; (2) control matrices for DQ assessment are suggested by the literature [27], and (3) they enable independent visual encodings of multiple structurally different aspects of checks as columns [23].

The sketch also illustrates a table that can be split into multiple parts, where corresponding rows are connected with a line. Similar to work by Gratzl et al. [10], the idea was to support sorting table parts independently. This feature, however, was not implemented due to feedback that considered it overly complex in the application context.

Semantically meaningful hierarchy of checks: Inspired by the well-known concept of data pivotization [40], the initial design already includes a user-definable hierarchy of checks. A hierarchical definition of rows ensures the scalability regarding the number of checks (G3). It is also essential for a flexible summarization of check results (G2) and a scalable visual complexity (G4).

An early approach to defining the hierarchy was the column 'Hierarchical Clustering' (Fig. 6). Based on the idea of grouping indication patterns hierarchically by their similarity, we wanted to provide a structural overview without having to define a hierarchy in advance. However, this concept was not realized, as domain experts clearly preferred comparing semantically meaningful groups as rows.

Based on this feedback, we decided to enable a hierarchy definition by any categorical check properties, such as sensor types, problem classes, etc. (see Fig. 7a,b). This allows exploiting any known structure of dimensions as suggested by Turkay et al. [45], and enables our domain experts to reason about DQ on familiar aggregation lev-

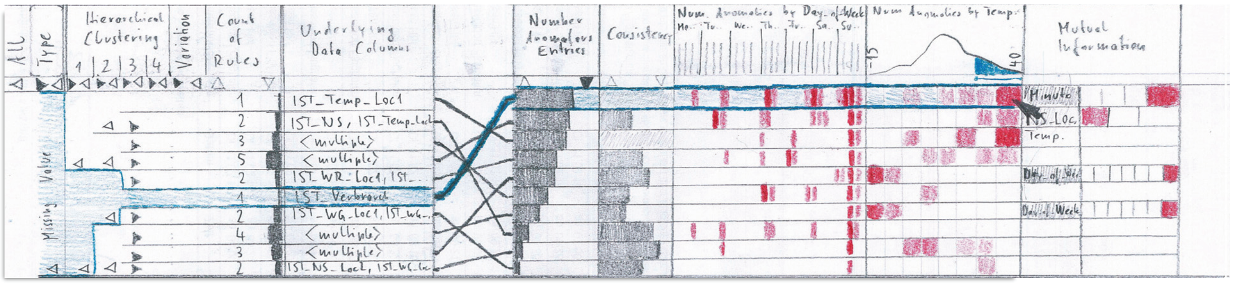


Fig. 6. A hand-drawn sketch of the DQ Overview showing parts of the initial design. Many sketched ideas persist to the final design, including the table-based layout, the hierarchical grouping of plausibility checks, or the concept of encoding different aspects of check results as columns.

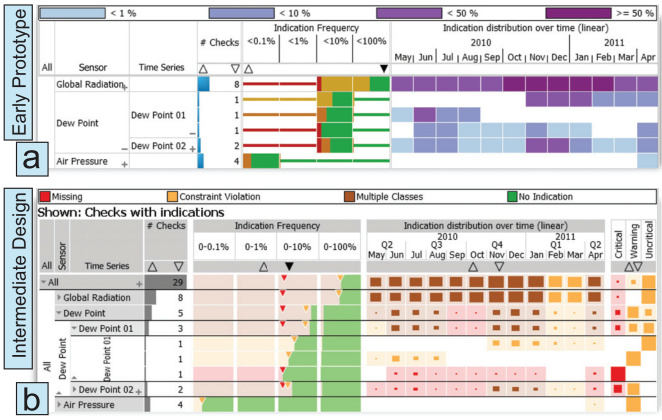


Fig. 7. The DQ Overview at different design stages. Aspects that evolved significantly over time include the encoding of the hierarchy (left), the use of color, the encoding of indication frequency (middle), and the encoding of indication distribution (right).

els, which they considered crucial to efficient DQ monitoring ***(LL01 - 'Preference for meaningful, familiar aggregation levels')***.

Encoding the hierarchy as indented layout: The hierarchy was initially designed as an icicle plot [25] (see Fig. 7a). Based on tests with real data, however, domain experts criticized the absence of sub-total rows. Expanded nodes could not be compared to non-expanded siblings, and child nodes could not be compared to their parents (see 'Dew Point 01' in Fig. 7a, which has no own representation).

We thus decided against icicle plots in favor of an indented layout with sub-totals, similar to a file browser (see Fig. 7b). Our domain experts preferred the indented layout for several reasons (***LL02: 'Expand rather than replace at drill-down***): (1) when expanding nodes, the parent was preserved as context, (2) less horizontal space was required for the hierarchy-part, and (3) the layout was more similar to well-known tree-widgets, e.g., file browsers. This could apply to hierarchical linear layouts in general (compare to Lex et al. [23]).

Following design guidelines for aggregation [8], we increased the discriminability of child nodes from aggregated parents by encoding the hierarchy level using shades of gray. To prevent users from unintentionally comparing rows of different hierarchy levels, we introduced black lines between them. Additionally, hovering a row desaturates all but that row and its child rows (see Fig. 3d). We found these visual cues necessary to preserve the visual correspondence between descriptions on the left side and columns to the far right.

Encoding indication frequency using scale-stack bars: Summarizing indication frequency as single value per row supports an efficient overview of overall DQ (G1). The initial sketch uses a simple linear encoding ('Number Anomalous Entries' in Fig. 6). However, tests with real data made us realize that indication frequencies exhibit largely varying scales. Using linear scaling, small percentages were not perceptible, while logarithmic scaling was considered harder to interpret. As a solution, we adopted the concept of scale-stack bar charts [14] and explicitly represented each order of magnitude as sub-column in decimal steps, i.e., $\leq 100\%, \leq 10\%, \leq 1\%$, etc. To enable an efficient selection of clean data, we explicitly visualized records without indications as a stacked green bar (see Fig. 7a).

The increased complexity of scale-stack bar charts raised usability concerns from some of our domain experts. We thus experimented with a number of variations. Fig. 7b shows one of our later attempts; all bars were drawn in full height, corresponding to bar charts with scale breaks [14]. The domain experts considered the resulting graph more visually pleasing. While it may also seem easier to interpret, it can be misleading, as adjacent bars can be perceived as a single long bar. In the last row of Fig. 7b ('Air Pressure'), the green bar is almost twice the size of the bar above ('Dew Point 02'), but the actual difference of clean records is less than 5%. After realizing this, experts who did indeed misinterpret the graph now indicated a preference for the scale-stack bars. We thus returned to the previous encoding, and visually reduced orders of magnitude below the smallest one containing the indication frequency value to a thin line (Fig. 2). Indication bars in larger magnitudes are also drawn in full height, which facilitates the comparison of indication frequencies within the same magnitude. Comparisons of magnitude can be performed effectively, as our design creates a staircase-like appearance with bars drawn in full height.

Consistent color encoding of check properties (e.g., 'Class'): In the initial design, color was used to encode relative indication frequencies in shades of red (see Fig. 6). We had not anticipated that certain meta-information should be displayed at all times, regardless of the hierarchical drill-down. From tests with real data, we discovered check severity as one such property, as any number of indications could be tolerable for one check, but critical for another. In the prototype, we thus introduced an additional scheme: following a familiar 'traffic light' metaphor brought up by a domain expert, we encoded the severity of indications using red, orange, yellow, and green for 'critical', 'warning', 'uncritical', and 'no indication' (see Fig. 7a).

As expected, hue allowed for an effective distinction of categorical meta-information [26]. However, feedback from companies at E-world indicated a need to encode different properties, and to customize color schemes according to internal conventions. We therefore decided to let users define the meta-information property used for coloring. Experimenting with different properties, our domain experts claimed that the problem class was typically at least as important as severity, which we adopted for the coloring in our guiding example. In general, hue is a suitable visual attribute for encoding qualitative information of at most 12 categories [48], which is typically the case for properties like 'Class'. At this point, we also decided to use the color encoding of the user-defined check property consistently for all parts of Visplause. Consequently, the distribution of severity levels or other properties was henceforth displayed in dedicated optional columns.

In hindsight, we had dedicated the color channel very early to indication frequency, a variable that could have been encoded more effectively using other means (e.g., area [26]). We also learned that inflexible use of color can be problematic in real-world projects, as customers may have internal policies or associations from other tools that prohibit certain uses. In this respect, encoding a customizable check property was a more sustainable choice (***LL03 - 'Consider workflow integration in color encodings'***).

Using a dedicated color for property overlap: With the consistent color-encoding of categorical check properties such as 'Class', we first attempted to convey the proportion of each category for all columns of the DQ overview. For example, the Indication Frequency Column subdivided the stacked bar by class. However, the perception of small bars could not be guaranteed, and the visual complexity was very high. Reflecting the principle of 'overview first, details on demand' [36] and following user feedback, we instead decided to use a dedicated category for overlap (e.g., 'multiple classes'), with on demand disambiguation via drill-down (***LL04: 'Increased simplicity by a dedicated category for overlap***). Visualizing aggregates based on multiple categories in a dedicated color reflects the guideline of 'visual simplicity' [8], while the 'fidelity' of the aggregated information is preserved by small triangles encoding the percentage of data records for each category in the Indication Frequency Column [8].

Encoding indication distribution using meaningful partitions: Indication Distribution Columns characterize the distribution of DQ problems with respect to different aspects. Their key idea of partitioning data records was already envisioned in the sketch (Fig. 6), where 'Num. Anomalies by Day of Week' shows a temporal partitioning into a weekly cycle, and 'Num. Anomalies by Temp.' a quantitative partitioning of temperature values. We also proposed an alternative scheme that defined one partition for each pixel, adjustable by resizing the column. However, our domain experts preferred a less fine-grained aggregation in terms of meaningful partitions for easier interpretation, e.g., months for temporal partitioning or intuitive steps for quantitative partitioning (***LL05: 'Meaningful partitions rather than maximal detail'***). This allowed partitioning schemes to add a dedicated partition for missing values in the underlying time series (see 'M' in Fig. 4c).

Encoding indication distribution by area: Until the prototype stage, relative indication frequency per partition was visualized using discrete steps of color intensity (see the color legend in Fig. 7a). However, readers of this visualization constantly referred to the legend to look up the encoded magnitudes. Following considerations on channel efficiency [26], we decided to linearly map relative indication frequency to the *area of centered rectangles*, with a minimal size of 2x2 pixels for small non-zero frequencies (see Fig. 7b). Additionally, a desaturated background supports the perceptual discrimination of segments with and without indications (i.e., those with a white background). We chose centered rectangle markers because they are automatically aligned both horizontally and vertically, allowing for efficient comparisons in both directions. The new encoding allowed us to consistently color rectangles by properties of the underlying checks (e.g., 'Class').

Supporting hypothesis generation in a linked view: In the sketch (Fig. 6), the 'Mutual Information' column shows which time series explains the indication pattern per row best. However, as the domain experts stressed the need for inspecting more than one time series for hypothesis generation, we started questioning whether an encoding of this information within the tabular layout was appropriate. Instead, we chose to address hypothesis generation using a linked 'Statistical Summary View' (Sec. 4.3.2), which provides two benefits: (1) It enables to explain the indication pattern for any subset of data records defined by selecting cells in the DQ overview, and (2) it shows whether multiple time series provide similar amounts of information gain.

Providing simple, dashboard-like entry points: A key insight from feedback at E-World concerned the configuration of Visplause. Many experts liked the flexible drill-down, but stressed the necessity of simple, well-chosen entry points for adoption (e.g., Fig. 3a). Previously, we started demonstrations with multiple columns and a drill-down by time series. A simpler and highly aggregated starting point with drill-down only on demand received much better acceptance by domain experts and was in line with the Information Seeking Mantra [36] (***LL06: 'Confusion of*** **overview first** ***with*** **overwhelm first*'***).

6 Implementation Aspects

Visplause is implemented in C++. It uses OpenGL for rendering and Gtk+ as GUI library. Different threads are responsible for rendering views, handling user input, and the evaluation of checks, ensuring that costly anomaly detection algorithms do not freeze the system [29].

Data Management: As outlined in Fig. 1, time series are represented in Visplause as a multivariate data table (i.e., time stamps as entities), plus meta-information defining properties per time series. Both can be imported from CSV files, or from a database via ODBC, which is typically preferred when monitoring regularly acquired data. Expert users can also modify and extend the meta-information at run-time.

Check Management: We discriminate between two use cases for check specification and evaluation: (1) Checks are created and evaluated in external data management or data warehousing systems. In practice, this is often the case and needs to be supported for an integration into existing workflows and systems (G5). To this end, Visplause allows the import of pre-evaluated checks as additional boolean-valued time series. (2) Checks are defined within Visplause. For this purpose, users may create and modify check rules in the script languages R, Matlab, or Python. Scripted rules comprise a user-defined script as well as a mapping for synchronizing time series within Visplause with named variables in the workspace of the script language before and after the execution of the script. These custom check rules can be applied to time series to instantiate plausibility checks. For example, applying the rule 'non-zero at night' to the time series PV_01, PV_02, and PV_03 creates three concrete checks. Scripted checks are also useful to leverage advanced algorithms for anomaly detection as provided by existing software packages in languages such as R. Parameters of user-defined checks, such as the script or the meta-information, can be modified at any time, which triggers a re-evaluation of the check and updates all views. Further details on the script integration are beyond the scope of this paper.

7 User Feedback

In addition to feedback collected during the design process (Sec. 5), Visplause has been deployed for one month to five domain experts from two companies in the energy sector, i.e., a transmission system operator and an IT-solution provider (see Sec. 2.1). The users include experts tasked with analysis and reporting in asset management, the classification of outage scenarios, or the optimization of control energy acquisition. All energy experts had several years of experience in their field. They do not have a strong background in visualization, however, some of them had used generic data exploration software before.

As observed by previous work [18], the domain experts confirmed DQ assessment as one of their most time-consuming tasks. Previously, DQ assessment had involved inspecting raw data tables, time series graphs, or sums of indications by user-defined plausibility checks in tools like Excel. As a result, the effort for a detailed DQ assessment was not considered justified in several cases.

After two hours of initial training, the experts used Visplause as part of their routine workflows, e.g., prior to statistical modeling. According to the experts, Visplause in general allowed them to conduct a thorough DQ assessment in a much shorter time. One user reported that DQ assessments that used to take two hours now only took half an hour to achieve the same level of confidence. Once a suitable set of checks had been defined, consulting Visplause was fast enough to precede any analysis. Besides gains of efficiency, this increased the confidence in the DQ for many cases which would previously not have been inspected in detail. One expert emphasized that Visplause enabled him for the first time an efficient DQ assessment of five million data records and dozens of time series.

As one negative aspect, the users stated that not all user interface elements for view configuration were intuitive to find, and creating hierarchies and column setups in the DQ Overview required some training. Once they obtained effective setups, however, they considered the visualization as suitable for presenting DQ problems to other stakeholders; the efficient overview and drill-down substantially facilitates the argumentation of DQ problems to data providers. In particular, they felt that creating check hierarchies along semantically meaningful categories allowed them to make use of their domain knowledge to structure checks in ways meaningful to them and other stakeholders. Additionally, they particularly liked the Indication Frequency Column. First, it allowed them to assess the overall DQ of entire data sets with a single glance. Second, they were able to visually compare the check results for a large number of checks, especially when sorted by indication frequency (which we have since made the default). Although they were unfamiliar with scale-stack bars, they liked our final design and found it easy to use and interact with.

The Experts particularly appreciated the indication-based selection of data subsets for downstream processing. We were surprised that their use of Visplause soon included overviews of script-based indicators in contexts beyond DQ. Some experts used checks to select representative training data. Tuning thresholds of scripted checks (see Sec. 6) enabled them to conduct a sensitivity analysis before feeding the resulting subset to a forecasting tool. According to these experts, the tight integration of check modification and result inspection yields a speed-up by a factor of ten over previous approaches for this task.

We have since, several months after the initial deployment, received further informal feedback from our users. Several of them have incorporated Visplause in their workflows and use it frequently. One partner plans to permanently install a Visplause-based dashboard for online monitoring of energy data on a large info-screen in the company.

8 Problem Abstraction

The design of Visplause was guided by the described tasks and goals in the context of the energy sector. However, showing prototypes to collaborators in other domains revealed that many aspects cover a much broader scope. As a recent example, we have deployed a Visplause-based application at an industrial manufacturing company since May 2016 (see supplemental material). In this application, dozens of quantities are measured for each manufactured item, e.g. material density, surface smoothness, etc. All of these quality measures must lie within pre-defined tolerance thresholds, otherwise, the item is discarded. Aside from daily quality control, a goal is to understand relationships between product deficiencies and parameters like the speed of production machines for process optimization (T2). For this application, the data model is a multi-dimensional table of manufactured items as rows. Attributes comprise a production timestamp, production parameters and the numerical quality measures. Checks are used to indicate items exceeding the quality tolerance limits. Based on meta-information about quality measures (e.g., the assessed part of the item, or the step in the production pipeline), Visplause provides a structured overview of the type of quality problem and supports breaking down the results by production timestamp or process parameters. Preliminary user feedback suggested that Visplause can effectively support process quality control as well as defect analysis. The lessons learned from the design process in the energy sector (Sec. 5) were considered directly transferable to the manufacturing application.

Comparing the data model and tasks of the manufacturing and energy contexts, the requirements of an application of Visplause can be abstracted along multiple dimensions: (1) As opposed to regular energy time series, the manufacturing data is a **multi-dimensional table** of items with numerical, categorical and temporal attributes. Checks do not rely on temporal semantics, and Indication Distribution Columns support numerical and categorical attributes as well (see Sec. 4). (2) Instead of sensor meta-information (e.g., type or location), the manufacturing application uses item parts or steps in the production pipeline assessed by quality measures. Technically, all **categorical properties about checks and underlying data attributes** can be used to structure the DQ Overview, regardless of the application context. (3) Instead of assessing energy time series with respect to DQ issues, the manufacturing application uses checks for thresholding production quality measures. In an abstract sense, checks can be seen as **evaluable statements computing a binary result** for each data value which reflects the presence of information with certain semantics.

We note that abstracting check results as sets of data records relates Visplause to approaches of set-typed data as surveyed by Alsallakh et al. [1]. However, most of the previous work focuses on representing intersections but disregards set provenance and semantics. Moreover, no set visualization so far fulfills the tasks and goals in Sec. 2.1.

As further examples from application domains beyond energy, we collected initial feedback from long-term partners and experts in automotive engineering. In their opinion, Visplause could provide an effective overview of user-defined constraint violations in the context of a multi-objective optimization based on parameter studies. For example, Indication Distribution Columns could show the distribution of constraint violations over varied simulation parameters. Finally, one expert from a collaboration in the healthcare sector suggested to use Visplause as an overview of key performance indicators (KPIs) in the context of business intelligence.

We thus see that the tasks and the data model of Visplause may be applicable to a broad scope of problems. Further deployments and evaluation for domains beyond energy are a key topic for future work.

9 Discussion and Future Work

DQ assessment is a topic of high practical relevance, as confirmed by literature and our own experience. Visualization is often mentioned as suitable for detecting DQ issues, however, little previous work addresses the routine assessment and validation of DQ issues for many regularly acquired time series. We thus regard Visplause as an approach that fills this gap and supports well-defined tasks.

A key benefit of Visplause over previous work concerns its scalability for large numbers of checks and underlying time series. The hierarchical structuring enables domain experts to reason about the DQ of many time series on semantically meaningful aggregation levels. The use of partition-based columns furthermore avoids clutter for any number of data records. As one limitation regarding scalability, our current implementation employs vertical scrolling for tables exceeding the available screen space. While scrolling is familiar to our users who did not consider this a draw-back, it contradicts the principle that eyes beat memory [26]. We thus intend to experiment with row-wise focus and context similar to the Table Lens [31].

While we consider design aspects of Visplause as the main contribution, the conceptual approach also follows the Visual Analytics Mantra [21]: We *analyze first* by evaluating plausibility checks, *show the important* by focusing on check indications, support to *zoom, filter, and analyze further* via the hierarchy and various columns, and show *details on demand* in linked views.

As one limitation of the data model, Visplause shows DQ problems only for existing data records, while semantic gaps in the data require an indirect specification via boundary records. Also, expressing indications in terms of data records is not always the most intuitive choice. For example, a check for implausible daily patterns should specify indications in terms of days rather than, e.g., hours. As future extension, we intend to allow various granularity levels for counting indications.

As another plan for future work, we intend to extend our approach for data cleansing, e.g., by imputing missing values. Information about data modifications fits our abstraction of checks and could easily be integrated within Visplause for overview and drill-down. Moreover, we plan to add animated transitions for interactions like drill-down and filtering. Finally, we plan to further evaluate Visplause by deployments in the healthcare sector, and to investigate applications outside DQ assessment.

10 Conclusion

This paper described Visplause, a new visualization approach for supporting the routine assessment of DQ based on automated plausibility checks. As a key benefit, Visplause provides a scalable overview of check indications with extensive possibilities for drill-down. Based on collaborations with partners from the energy sector, we identified key tasks and illustrated how Visplause supports them. We discussed design decisions and stated insights from a reflection of the design process in the hope that they are useful also in other application contexts.

Visplause was designed to meet requirements of DQ assessment. However, user feedback suggested a potential application also outside the context of DQ for providing an overview and drill-down of indicators in general. Motivated by the positive user feedback, we believe that Visplause can become an important approach for making DQ assessment more efficient and increasing the confidence in data. We will also investigate further applications of Visplause for enabling a guided analysis based on indicators.

Acknowledgments

This work has been supported by the Austrian Funding Agency (FFG) within the scope of the COMET K1 program and the FFG-funded project 843550 (DEXHELPP). Thanks go to all project participants of Hakom and Austrian Power Grid AG, and to O. Rafelsberger, S. Pajer, F. Endel, and E. Gröller for valuable comments.

REFERENCES

[1] B. Alsallakh, L. Micallef, W. Aigner, H. Hauser, S. Miksch, and P. Rodgers. Visualizing Sets and Set-typed Data: State-of-the-Art and Future Challenges. *Eurographics Conf. on Visualization (EuroVis)– State of The Art Reports*, pages 1–21, 2014.

[2] C. Batini and M. Scannapieca. Data Quality Dimensions. *Data Quality: Concepts, Methodologies and Techniques*, pages 19–49, 2006.

[3] M. Brehmer, J. Ng, K. Tate, and T. Munzner. Matches, Mismatches, and Methods: Multiple-View Workflows for Energy Portfolio Analysis. *IEEE Trans. on Vis. and Comp. Graphics*, 22(1):449–458, 2015.

[4] V. Chandola, A. Banerjee, and V. Kumar. Anomaly Detection: A Survey. *ACM Comput. Surv.*, 41(3):pages 15:1–15:58, July 2009.

[5] T. Dasu and T. Johnson. *Exploratory Data Mining and Data Cleaning*. John Wiley & Sons, Inc., New York, NY, USA, 1 edition, 2003.

[6] S. P. Dow, A. Glassco, J. Kass, M. Schwarz, D. L. Schwartz, and S. R. Klemmer. Parallel Prototyping Leads to Better Design Results, More Divergence, and Increased Self-efficacy. *ACM Trans. Comput.-Hum. Interact.*, 17(4):pages 18:1–18:24, Dec. 2010.

[7] C. Eaton, C. Plaisant, and T. Drizd. Visualizing Missing Data: Graph Interpretation User Study. In *Proc. of the 2005 IFIP TC13 Int. Conf. on Human-Computer Interaction*, INTERACT'05, pages 861–872, Berlin, Heidelberg, 2005. Springer-Verlag.

[8] N. Elmqvist and J.-D. Fekete. Hierarchical Aggregation for Information Visualization: Overview, Techniques, and Design Guidelines. *IEEE Trans. on Vis. and Comp. Graphics*, 16(3):pages 439–454, 2010.

[9] S. J. Fernstad and R. C. Glen. Visual Analysis of Missing Data - To See What Isn't There. In *Poster Proc. of IEEE InfoVis*, 2014.

[10] S. Gratzl, N. Gehlenborg, A. Lex, H. Pfister, and M. Streit. Domino: Extracting, Comparing, and Manipulating Subsets across Multiple Tabular Datasets. *IEEE Trans. on Vis. and Comp. Graphics*, 20(12):2023–2032, 2014.

[11] T. Gschwandtner, W. Aigner, S. Miksch, J. Gärtner, S. Kriglstein, M. Pohl, and N. Suchy. TimeCleanser: A Visual Analytics Approach for Data Cleansing of Time-Oriented Data. In *Proc. of the 14th Int. Conf. on Knowledge Technologies and Data-driven Business*, pages 18:1–18:8, 2014.

[12] T. Gschwandtner, J. Gärtner, W. Aigner, and S. Miksch. A Taxonomy of Dirty Time-Oriented Data. In *Multidisciplinary Research and Practice for Information Systems*, pages 58–72. Springer, 2012.

[13] J. M. Hellerstein. Quantitative Data Cleaning for Large Databases. *United Nations Economic Commission for Europe (UNECE)*, 2008.

[14] M. Hlawatsch, F. Sadlo, M. Burch, and D. Weiskopf. Scale-stack Bar Charts. In *Eurographics Conf. on Visualization*, pages 181–190, 2013.

[15] K. Holtzblatt and S. Jones. *Contextual Inquiry: A Participatory Technique for System Design*, pages 177–210. Lawrence Erlbaum Associates, Hillsdale, 1993.

[16] IBM Corporation. IBM Watson Analytics. `http://www.ibm.com/analytics/watson-analytics`.

[17] H. Janetzko, F. Stoffel, S. Mittelstädt, and D. A. Keim. Anomaly Detection for Visual Analytics of Power Consumption Data. *Computers & Graphics*, 38:pages 27–37, 2014.

[18] S. Kandel, J. Heer, C. Plaisant, J. Kennedy, F. van Ham, N. H. Riche, C. Weaver, B. Lee, D. Brodbeck, and P. Buono. Research Directions in Data Wrangling: Visualizations and Transformations for Usable and Credible Data. *Information Visualization*, 10(4):pages 271–288, 2011.

[19] S. Kandel, R. Parikh, A. Paepcke, J. M. Hellerstein, and J. Heer. Profiler: Integrated Statistical Analysis and Visualization for Data Quality Assessment. In *Proc. of the Int. Work. Conf. on Advanced Visual Interfaces*, pages 547–554. ACM, 2012.

[20] D. A. Keim. Information Visualization and Visual Data Mining. *IEEE Trans. on Vis. and Comp. Graphics*, 8(1):pages 1–8, 2002.

[21] D. A. Keim, F. Mansmann, J. Schneidewind, J. Thomas, and H. Ziegler. *Visual Analytics: Scope and Challenges*. Springer, 2008.

[22] W. Kim, B.-J. Choi, E.-K. Hong, S.-K. Kim, and D. Lee. A Taxonomy of Dirty Data. *Data mining and knowledge discovery*, 7(1):pages 81–99, 2003.

[23] A. Lex, N. Gehlenborg, H. Strobelt, R. Vuillemot, and H. Pfister. UpSet: Visualization of Intersecting Sets. *IEEE Trans. on Vis. and Comp. Graphics*, 20(12):pages 1983–1992, 2014.

[24] LUMA Institute. Vision Statement: A Taxonomy of Innovation, 2014.

[25] M. J. McGuffin and J.-M. Robert. Quantifying the Space-Efficiency of 2D Graphical Representations of Trees. *Information Visualization*, 9(2):pages 115–140, 2010.

[26] T. Munzner. *Visualization Analysis and Design*. CRC Press, 2014.

[27] E. M. Pierce. Assessing Data Quality with Control Matrices. *Communications of the ACM*, 47(2):pages 82–86, 2004.

[28] L. L. Pipino, Y. W. Lee, and R. Y. Wang. Data Quality Assessment. *Communications of the ACM*, 45(4):pages 211–218, 2002.

[29] H. Piringer, C. Tominski, P. Muigg, and W. Berger. A Multi-Threading Architecture to Support Interactive Visual Exploration. *IEEE Trans. on Vis. and Comp. Graphics*, 15(6):pages 1113–1120, 2009.

[30] E. Rahm and H. H. Do. Data Cleaning: Problems and Current Approaches. *IEEE Data Eng. Bull.*, 23(4):pages 3–13, 2000.

[31] R. Rao and S. K. Card. The Table Lens: Merging Graphical and Symbolic Representations in an Interactive Focus+Context Visualization for Tabular Information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 318–322. ACM, 1994.

[32] S. Sadiq, N. K. Yeganeh, and M. Indulska. 20 Years of Data Quality Research: Themes, Trends and Synergies. In *Proc. of the Twenty-Second Australasian Database Conf.*, volume 115, pages 153–162, 2011.

[33] M. Sedlmair, M. Meyer, and T. Munzner. Design Study Methodology: Reflections from the Trenches and the Stacks. *IEEE Trans. on Vis. and Comp. Graphics*, 18(12):2431–2440, 2012.

[34] J. Seo and B. Shneiderman. A Rank-by-Feature Framework for Interactive Exploration of Multidimensional Data. *Information Visualization*, 4(2):pages 96–113, 2005.

[35] L. Shi, Q. Liao, Y. He, R. Li, A. Striegel, and Z. Su. SAVE: Sensor Anomaly Visualization Engine. In *IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 201–210. IEEE, 2011.

[36] B. Shneiderman. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *Proc. of the 1996 IEEE Symposium on Visual Languages*, pages 336–343, 1996.

[37] M. Steiger, J. Bernard, S. Mittelstädt, H. Lücke-Tieke, D. Keim, T. May, and J. Kohlhammer. Visual Analysis of Time-Series Similarities for Anomaly Detection in Sensor Networks. *Computer Graphics Forum*, 33(3):pages 401–410, 2014.

[38] R. Sulo, S. Eick, and R. Grossman. DaVis: a Tool for Visualizing Data Quality. *Posters Compendium of InfoVis*, pages 45–46, 2005.

[39] D. F. Swayne, D. T. Lang, A. Buja, and D. Cook. GGobi: Evolving from XGobi into an Extensible Framework for Interactive Data Visualization. *Computational Statistics & Data Analysis*, 43(4):pages 423–444, 2003.

[40] Tableau Software. Tableau. `http://www.tableau.com`.

[41] Talend Inc. Talend. `https://www.talend.com`.

[42] M. Templ, A. Alfons, and P. Filzmoser. Exploring Incomplete Data using Visualization Techniques. *Advances in Data Analysis and Classification*, 6(1):pages 29–47, 2012.

[43] M. Theus, H. Hofmann, B. Siegl, and A. Unwin. MANET Extensions to Interactive Statistical Graphics for Missing Values. In *New Techniques and Technologies for Statistics II*, 1997.

[44] TIBCO Spotfire. Spotfire. `http://spotfire.tibco.com`.

[45] C. Turkay, A. Lundervold, A. J. Lundervold, and H. Hauser. Representative Factor Generation for the Interactive Visual Analysis of High-Dimensional Data. *IEEE Trans. on Vis. and Comp. Graphics*, 18(12):2621–2630, 2012.

[46] R. Y. Wang and D. M. Strong. Beyond Accuracy: What Data Quality Means to Data Consumers. *Journal of Management Information Systems*, pages 5–33, 1996.

[47] M. Ward, Z. Xie, D. Yang, and E. Rundensteiner. Quality-Aware Visual Data Analysis. *Computational Statistics*, 26(4):pages 567–584, 2011.

[48] C. Ware. *Information Visualization: Perception for Design, Third Edition*. Morgan Kaufmann, 2012.

[49] B. W. Wong and M. Varga. Black Holes, Keyholes and Brown Worms: Challenges in Sense Making. In *Proc. of the Human Factors and Ergonomics Society Annual Meeting*, pages 287–291, 2012.