Natto : Rapid Visual Iteration of Analytic Data Models with Intelligent Assistance

Anamaria Crisan and Vidya Setlur

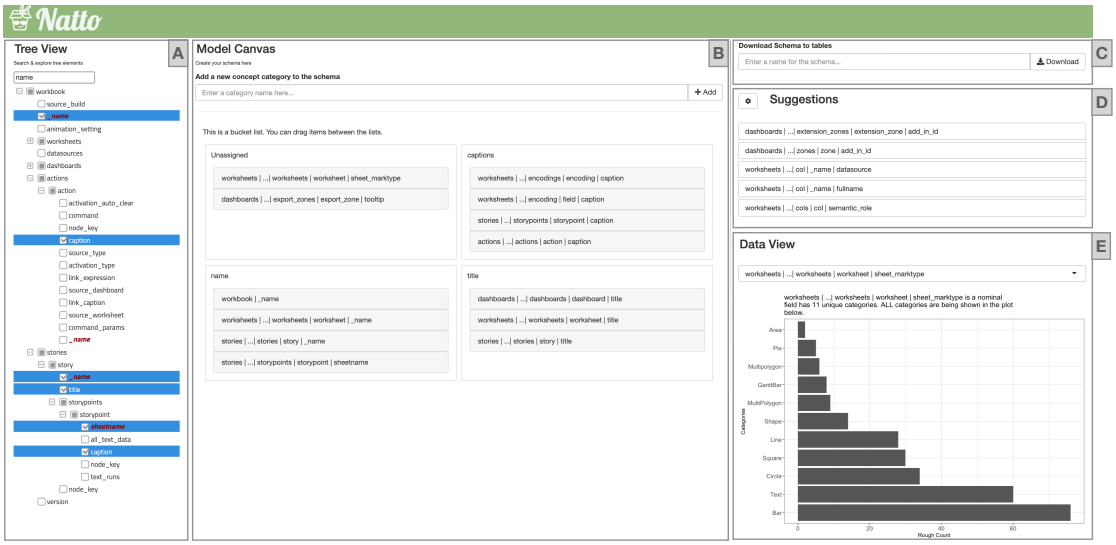

Fig. 1. *Natto* Interface. **A** - The Tree Viewer Pane shows a consensus tree from multiple data sources. **B** - The Model Canvas Pane shows an analytic data model that the user is in the process of constructing. The nodes selected in the tree are added to the Model Canvas Pane and shuffled between user-defined concept cards. **C** - An export option to download the analytic data model and its corresponding data. **D** - Automatically generated suggestions for nodes to include in the conceptual model that correspond to the contents of the Model Canvas Pane. **E** - Univariate visualizations of selected data containing leaf nodes.

Abstract—Data analysts need to routinely transform data into a form conducive for deeper investigation. While there exists a myriad of tools to support this task on tabular data, few tools exist to support analysts with more complex data types. In this study, we investigate how analysts process and transform large sets of XML data to create an analytic data model useful to further their analysis. We conduct a set of formative interviews with four experts that have diverse yet specialized knowledge of a common dataset. From these interviews, we derive a set of goals, tasks, and design requirements for transforming XML data into an analytic data model. We implement *Natto* as a proof-of-concept prototype that actualizes these design requirements into a set of visual and interaction design choices. We demonstrate the utility of the system through the presentation of analysis scenarios using real-world data. Our research contributes novel insights into the unique challenges of transforming data that is both hierarchical and internally linked. Further, it extends the knowledge of the visualization community in the areas of data preparation and wrangling.

Index Terms—Data Modeling, Data Analysis, Hierarchical Data

1 Introduction

Data scientists and business analysts routinely transform data into a suitable shape for answering their pertinent analytic questions [6].The data is often stored as pre-defined data models that are developed to support organizational practices [15, 17]. However, these pre-defined data models do now always align with needs of these individual data scientists and analysts, thus necessitating the data be further transformed for each analytic question [6]. When undertaking these data model transformations, analysts are confronted with the challenge of *both* understanding the existing structure of the data [32] and modifying it to appropriate transformations [38]. This transformation process is often iterative and requires the analyst to construct and consider multiple alternatives before arriving at a model conducive for data analysis [37]. The challenges of carrying out these data transformations are especially cumbersome for non-tabular data types that form a smaller, but still sizable portion of business data [41]. In addition, few data analysis tools and visualization systems provide support for non-tabular data [5, 54].

We observed these challenges while working with domain experts seeking to analyze data from XML documents. Compared to tabular datasets, XML data and other similar data types like JSON or YAML, contain pertinent hierarchical and internally linked data relationships. These additional properties can make the process of understanding, transforming, and evaluating data, complex. Motivated by these challenges, we conducted a formative design study with these domain experts to better understand the challenges they face, their objectives in their analyses, and to identify a set of design requirements to address their unmet needs. We set out to surface how analysts created these 'analytic data models' from existing data models to address a specific data question. We especially highlight the challenges of conceptualizing what form the data must take, relative to existing structure, in order for an analyst to achieve their analysis goal [35].

• *A. Crisan and V. Setlur are with Tableau Research {acrisan, vsetlur}@tableau.com*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

The goal of our design study is to investigate how analysts transform XML data or more generally data that is hierarchical and internally linked, into an analytic data model. We were especially interested to capture these transformations from a common initial dataset in order to examine if and how their strategies for preparing and transforming these data differed. We conducted interviews with four domain experts that were using a common dataset of XML documents to conduct analyses for different data questions.

We present three contributions in this paper. First, we provide a working definition of analytic data models and differentiate from that of existing data models. Second, we present a formative design study that identifies the goals, tasks, and design requirements for understanding how an existing XML based data model can be transformed into an analytic data model. Third, we present the *Natto* prototype that enables visual and interaction design choices for iteratively creating analytic data models from XML (and other hierarchical) data. Our research brings together existing techniques from databases, knowledge engineering, and visualization research to present a novel user experience for transforming data that is both hierarchical and that also contains linkages *within* this hierarchical structure.

1.1 Existing vs. analytic data models

Our research highlights how analysts must routinely explore an existing data model ahead of carrying out data transformations to prepare their data for analysis [6]. We referred to the former as **existing data models**, which largely serve organizational needs, while the latter are **analytic data models**, which address a specific data question.The existing data models are reified from a pre-determined schema established to support organizational practices [15, 17]. Analytic data models manifest at the end of a series of transformations from the existing data model. While the importance of carrying out these transformations is understood in the research literature [16, 29, 47, 59], there has been a paucity of research that addresses the needs of analysts concerning non-tabular data. Existing research also does not explore the tension that exists when data models do not align with individual analysis goals.

We argue that such types of transformation from one existing data model to a new one more suitable for analysis (the analytic data model), are not well understood or supported by existing data preparation and wrangling tools. Here, we highlight this data model transformation process and in particular, the conceptual work the analyst must perform to develop this analytic data model. We do so by describing the challenges, approaches, and failures that a domain experts faced when examining a common dataset of XML documents and attempting to derive analytic data models for different data questions. Even while analysts with programmatic capabilities were able to extract, transform, and load a new data model into their analysis tools, the process was often tedious and time-consuming. One particular challenge of the transformation process and one we see as being overlooked by existing tools, is the conceptual component where the analyst plans that delineation between the existing data model to a new data model for analysis. This conceptual component requires that the analyst considers *both* the data and how it should be shaped relative to some analytic goal. Our use of the term 'analytic data model' is intended to capture both the conceptual and data transformation processes involved in exploring, conceptualizing, extracting, and transforming data from an existing data model into an analytic form.

2 RELATED WORK

We discuss prior research in data modeling, data preparation, and tools for visualizing hierarchical data.

2.1 Data Modeling

Data modeling is fundamental to the process of analysis and design methodologies for the development of information systems [58]. Conceptual modeling facilitates the early detection and correction of system development errors and is often performed prior to data analysis [26,63]. Conceptual modeling formalism such as Entity-Relationship (ER) modeling [14] is commonly used for structured analysis. The ER approach provides an intuitive model of entities and their relationships, allowing for translation into a database schema [19]. However, relational data models can be inflexible to changes in data [24] and struggle with more complex data relationships including hierarchical [49] and linked data [33]. Other research in conceptual modeling focuses on the use of ontologies for developing, comparing, and improving data models [8, 53, 62]. Mayadewi *et al.* [40] explored the process of transforming a relational database into an ontology model by extracting hidden semantics from a relational model.Ontologies share common canonical structure and properties to hierarchical markup such as XML, especially around conceptual modeling, inheritance, and inference mechanisms [20, 33].

The simplicity and flexibility of XML has led to its adoption for providing a broad range of high quality information. Relational databases (RDB) are a common mechanism for storing and querying the content of XML data. Researchers have proposed techniques to map XML data into a relational schema of an RDB [43, 60]. Further improvements involve automation [36] and developing more performant techniques [50]. Boukottaya and Vanoirbeek [9] focus on XML document schema matching and its reuse for mapping transformations. Others have developed techniques [42, 51] where domain experts could reverse engineer data-centric XML schema from their conceptual models so that changes can be propagated for consistency.

Our research explores ways that analysts can use visual interfaces to specify an analytic data model. We posit that visual interfaces are an effective way to allow individuals both with and without knowledge of the data model to be able to intuitively transform the model for analysis. Specifically, we build on prior work to help analysts explore the structure and content of existing data and use this knowledge with automated support.

2.2 Data Preparation

Data preparation (prep) is often an arduous process for transforming data into a form suitable for analysis and is underserved by existing visualization and analysis tools. [30, 48]. Various ideas have been proposed that include visual approaches to aid with data wrangling and integrate with automated approaches to allow an interactive editing cycle [21, 28, 38]. These techniques become more complicated with complex data ingest pipelines [57].

Several works of research have focused on developing tools to help users with various aspects of the prep process. Zernichow and Roman [61] developed a data profiling tool that identifies and visualizes potential data quality issues, relevant data cleaning functions, and an interactive spreadsheet view. Potter's Wheel [45] allows users to gradually build transformations to clean the data by adding or undoing transforms in a spreadsheet-like interface. Wrangler [29] combines direct manipulation of visualized data with automatic inference of relevant transforms, enabling analysts to iteratively explore the space of applicable operations and preview their effects. Other wrangling-by-example systems include Foofah [27] and recent work on multi-table wrangling [31]. While these tools focus on tabular data, people increasingly analyze non-tabular data, including time-oriented [3, 23], traffic [1], and network data [5, 25, 39, 55]. There are also methods that exist for flagging statistical anomalies in electronic dictionaries stored in XML format [7]. Other work automates the wrangling process by incorporating data context in the target schema focusing on mapping validation, value format transformation, and repair [34].

Our research extends this work by exploring data transformation applied to XML data and the process of transforming from a hierarchical to a tabular data representation.

2.3 Exploring XML and other Hierarchical Data

A relevant aspect to our work is the suite of software tools for exploring XML schema through visual user interfaces. Previous research has focused on the composition relationships in an XML Schema document by providing a tree view that displays related data types and elements [10, 12]. Data transformation problems are very common and are challenging to implement for large and complex datasets. Robertson et al. [46] demonstrated the use of visualization techniques to help users

deal with schema matching, particularly for large, complex information. Clip [44] is an XML Schema mapping tool where the mappings explicitly specify structural transformations in addition to value couplings. Other work described an approach for specifying data mapping transformations between XML schemas using a combination of automated schema analysis and selective user interaction [2]. Recently, visualization tools have focused on representing hierarchical information such as ontologies using indented lists, node-link graphs, Euler diagrams, and tree maps. Vispedia* is an early example of a tool where models shared data as a semantic graph to enable search and discovery of datasets with different underlying data models [13]. CAVA similarly constructs a knowledge graph over relationship data to augment data analysis [11]. The tools display hierarchical relationships between entities, showing details on demand [18]. Other work explored multiple coordinated views for displaying ontologies in XML format [52].

A limitation of these existing tools is that they do not address scenarios where the analyst must *both* explore and transform data from one data model format to another. We implement a prototype tool, *Natto* that tackles this problem most directly, while taking inspiration from prior systems, Vispedia* and CAVA, to create a visual and interactive user experience.

3 Domain Context and Data Description

Our design study was carried out in collaboration with domain experts, all at the same business intelligence technology company. In this section, we describe their domain context and the characteristics of their XML documents. We also present a usage scenario to articulate what is captured by the XML data model and how it can be analyzed.

Domain Context and Challenges. We worked with experts that needed to analyze stored data pertaining to a set of XML documents in the visual analysis tool. The existing data model for the XML documents was defined by software engineers and as such was optimized to support the loading of these documents into the tool. When an end-user uses this visual analysis tool to conduct an analysis, the data are stored within this XML document. The types of data stored varies from automatically generated system settings (for example, the software version, the sizes of elements in the display) as well as user generated content (for example, the specific fields being visualized and the way the end-users choose to visualize them). An analyst seeking to examine how end-users use the tool, can gather and analyze these documents, extracting data relevant to a particular question. For domain experts that we worked with, the data that was especially interesting to them was user-generated content, which tended to be stored in leaf nodes of the existing data model.

A particular feature of this XML document was that it contained redundant and nested elements to store data. Fig. 2 illustrates an existing XML data model with redundant subtrees encoding similar elements (worksheets), but storing different data. For example, an end-user seeking to visualize their data using this visual analysis tool can create several worksheets, each showing a different slice and visual representation of the data. On the first worksheet, they might compare the prices of two hypothetical widgets *A* and *B* using a line chart. On a second worksheet, they might calculate and evaluate the total production of just widget *A* over time. These actions are captured and stored in the XML document as separate sub-trees within the XML document. We found that this tree structure made it difficult for analysts to explore, extract, transform, and analyze data. Simply parsing the XML document proved not be enough; the transformations that analysts had to undertake to make the data usable for analysis, went beyond beyond operations like joining, filtering, grouping, or other transformations [28]. The sources of these challenges were two-fold. First, it was not easy to explore the existing data model and tools like XML viewers were too limited to address their needs. For smaller documents containing a limited amount of data, this issue is not substantial. However, the XML documents that our domain experts had to analyze, were large and dense tree because they are intended to support a fairly complex piece of visual analysis software. Second, the hierarchical schema representation made it difficult to quickly apply analytics techniques from standard analytic tools, such as Microsoft Excel or Tableau, or programmatic libraries, such as Pandas [64] or the tidyverse suite [65].

3.1 Data Description and Abstraction

Existing Data Model Structure. The existing data model defined by XML schema is a tree representation, containing interior and leaf nodes (Fig. 2). This tree representation is also internally linked, meaning that a common data field appears in multiple nodes throughout the tree. More specifically, the visual analysis tool has multiple worksheet interfaces where people can add, view, and modify data leading to multiple nodes in the underlying XML structure. Within the XML structure, these worksheets are represented as parallel and separate subtrees, but these subtrees can also have relationships between them facilitated by overlapping data in the worksheets (Fig. 2). During the study, experts wished to navigate this tree, identifying and reorganizing both interior and leaf nodes to support their data analyses. While subtree relationships were interesting to the domain experts, they were tedious to discover and difficult to extract with existing analysis tools or libraries.

Data Model Consistency. An important consideration of working with XML data concerns changes to the tree structure over time. The changes naturally occur as the software changes either because new features are added or because better and more efficient ways to store the data emerge. These changes to the data model over time are not unique to XML, but apply to relational data models as well. However, the tree structure of XML data models introduces new challenges that pertain to harmonizing this hierarchical structure over time [44, 46]. We did not have to address this particular issue in our work as we had at our disposal tools that were internally developed to ameliorate these differences in this data model over time. As a result, we were able to work with a common tree format in this design study. We also note that challenges stemming from normalizing XML documents into a common format is an ongoing and challenging research problem that we do not address.

4 Analyst Exemplars

In this section, we describe the objectives, process, and pain points in the exemplars solicited from interviews with four domain experts, henceforth referred to as analysts. These analysts add specificity to the fictitious usage scenario we described in the previous section. We detail the interview process and how this informed our subsequent goals, tasks, and requirements, as well as the visual and interactions design choices in the *Natto* prototype.

Overview. We identified four individuals, two that were actively analyzing the data and two others that had attempted to do so, but hit a failure mode and gave up. We arranged an initial interview asking them to describe their data questions and then provide a walk through of their process of identifying, extracting, and shaping data for analysis.We did not have a specific interview protocol, but took screen recordings and kept notes of our sessions with participants. Each session was approximately an hour in length. For individual's that reached failure modes, we sought to understand what those barriers were. Exemplars I and II represent individuals that had knowledge of the existing XML data model and had specific analytic questions. Exemplars III and IV represent individuals that had little knowledge of the existing data model, but wanted to explore the data to assess whether it was suitable for their more general analysis goals. These latter two exemplars reached failure points from their lack of familiarity with the existing data model. Exemplars also vary with respect to the crispness of the individuals objectives, with Exemplar I presenting the most crisp objectives and Exemplar IV presenting the least. From these sessions, we derived `tasks` for preparing and transforming data.

4.1 Exemplar I: Examining Text Content

A senior data scientist sought to investigate user-generated text content. They had no prior knowledge of the existing XML data model, but had invested the time to get familiar with the data model. They `created a visual overview` of the data using Code Beautify XML viewer

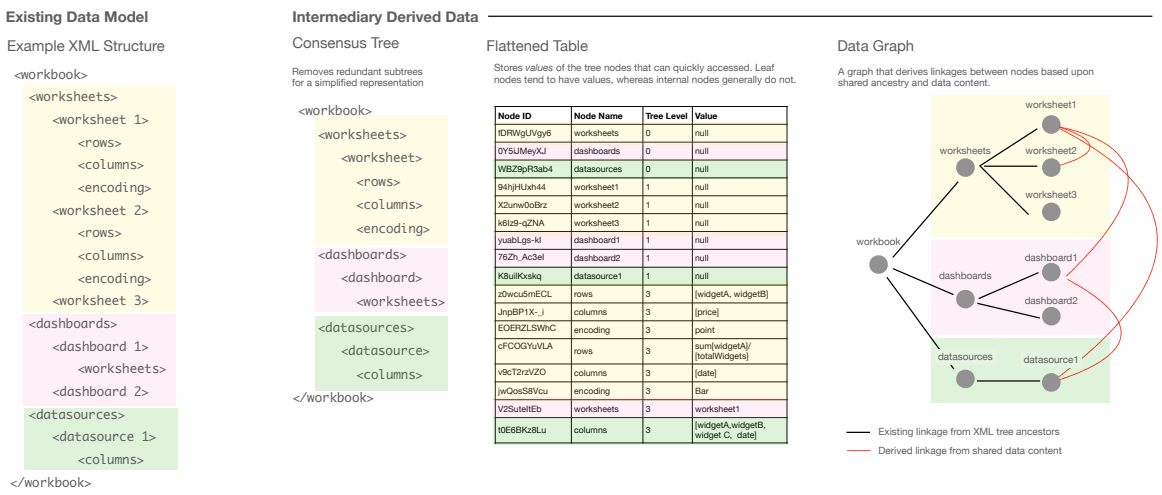

Fig. 2. Illustration of XML data and intermediary derived by *Natto*to support the creation of analytic data models.The XML data that we consider, contains many repeated elements (i.e. multiple worksheets),making it a slow processes for an analyst to traverse the XML Tree to extract and re-organize data for analysis. To expedite the analyst's workflow, *Natto* derives three data structures from the XML data: a consensus tree, a flattened data table, and a graph representing content similarity between nodes of the consensus tree. The derived data is used to simplify the analysts' navigation of the data hierarchy and for *Natto* to provide suggestions to the analyst.

(`https://codebeautify.org/xmlviewer`) and would `browse` or `look up` tags and attributes using the tool's search feature. As they examined the tree, they used a generic text editor to `keep track` of useful nodes they `identified`. They would subsequently `examine` a random sample of XML document in Python, `extracting data` from the nodes they had previously identified. They then performed a visual assessment using univariate data distributions. They further experimented with this data using various machine learning techniques to cluster the data. As they reviewed the results of the machine learning modeling, they made further revisions to the analytic model, which included identifying new nodes of interest, extracting the data, and reapplying their machine learning analysis. This iterative flow progressed through several cycles and in the process, the analyst `sketched` an evolving analytic data model. They subsequently used this analytic data model to transform from the existing XML structure to a new tabular format for machine learning analysis.

We observed several pain points in this process. The first, was the difficulty of stitching between different tools and libraries to view the data, extract, and analyze. Second, was the challenge of orienting themselves within the data in order to develop an analytic data model.

4.2 Exemplar II: Exploring User Strategies

A research consultant sought to ascertain how often users created multiple linked views when using the visual analysis tool. This individual was much more familiar with the existing XML data model compared to Exemplar I. However, they employed similar strategies, including the use of a XML browser, in this case Microsoft Notepad, to have a `visual overview` of the data. They also used the viewer to `browse`, `identify`, and `look up` data of interest. They extracted the data and used Tableau Desktop, their analysis tool of choice, to `view` and `examine` its contents and similarly produced univariate graphical summaries. However, this individual also summarized this data into a dashboard showing the connections between different nodes of the XML tree they were interested in. The dashboard organization not only helped `orient` them but helped to `visualize connections` between related content such as data about the visual encodings. We also noted that the dashboard content was grouped and organized to resemble the analyst's mental model of the data, for example, they organized content from different parts of the XML tree to summarize the visual encodings created by different end-users and the ways that they were linked.

Exemplar II experienced similar pain points as Exemplar I regarding the movement of data between multiple tools and orienting within the existing data model. In both instances, the ability to visualize and examine the existing model was important, as was the ability to progressively sketch out the analytic data model.

4.3 Exemplar III: Assessing the Breadth of Test Coverage

A senior product manager wanted to explore additional sources of data for assessing the breadth of their feature team's test coverage for the visual analysis tool. They wished to make a quick assessment before investing additional time and resources into this problem. They made initial attempts to examine the XML data on their own, but ultimately hit a roadblock. Instead, the exemplar relied on another individual to help them identify and extract data that they needed. Unlike the prior exemplars, their data question was more open-ended and exploratory, reflecting an "I'll know it when I see it" mentality, rather than the desire to identify and examine some specific aspect of the data. Still, their objectives were to `browse` and `identify` useful data within the XML documents to inform a conversation with their development team.

This exemplar revealed pain points surrounding the 'cold start' problem - while they had a rough goal in mind, they did not know where to begin to assess and transform the data model. We also surfaced the pain point of requiring someone who was versed in the data to orient them and potentially identify points of interest.

4.4 Exemplar IV: Data Provenance

A senior manager wanted to modify the existing data model in order to better support a feature of the visual analysis tool. They wanted to have a sense of how this modification could be incorporated into the existing data model and also to analyze some of its possible downstream effects on end-users. They were only slightly familiar with the existing data model and efforts to make changes, but had not extensively explored this model on their own. Similar to Exemplar III, their data question was fluid and depended on an evolving understanding of the data. They wished to make a similar `quick assessment` of whether the dataset could answer a particular question and if the underlying data could be used to further support ideation on change provenance more generally. Like the previous exemplars, they needed to easily `orient` themselves in the existing data model, perform `look-ups` for the proposed changes, and `explore` and `analyze` potential side effects of enacting such a change. We concluded that the pain points for this exemplar were similar to that of Exemplar III.

5 TASKS, GOALS, AND DESIGN REQUIREMENTS

From the exemplars, we derive a set of design goals to help inform the implementation of *Natto*. We particularly focused on goals (`G#`) and associated sets of tasks (`T#`) that help support the creation of an

analytic data model from an existing data model that is hierarchical and internally linked.

G1: Understanding and exploring existing data models. All exemplars began with an **`overview` (T1)** of the existing data model represented by the XML hierarchical tree structure to inform their analyses. This overview helped them to understand the type of data that was available or absent for analysis. In Exemplars I and II, the overview led to a targeted exploration of the hierarchical structure to **`lookup` (T2)** and **`identify` (T3)** useful data. The exploratory processes of Exemplars III and IV were oriented toward serendipitous **`browsing` (T4)** of the existing data model to triage potentially useful data. Critically, all individuals needed support to **`orient` (T5)** themselves within their exploration of the existing data model and as they created their analytic data models. We note that while analysts found the overview of the existing data model helpful, they did not need to interact with a facsimile of it. In fact, it would be helpful for them to interact with a version that removes redundant or non-informative tree elements.

G2: Creating a meaningful representation of an analytic data model. Exemplars I and II presented the strongest evidence that experts iteratively create an analytic data model as they explored the existing data model. From the existing data model they **`progressively added nodes` (T6)** from the XML tree into some scratch environment, actual sketches in Exemplar I, or a dashboard in Exemplar II. Over time, they would **`group these nodes` (T7)** into some semantically meaningful concepts that aligned with their data questions and analytic objectives. Exemplars III and IV struggled to create an analytic data model on their own; they relied on conversations with an intermediary to understand the existing data model and describe how they would need the data transformed for their data questions. Across all of these exemplars, **`surfacing links` (T8)** between nodes of the data hierarchy was important and also difficult. An example of this is the analyst in Exemplar I who sought to surface and extract text data from the existing data model. It would be helpful if there was automated support for this process to remove their current manual exploration. This latter task is tightly coupled with data orientation (T5).

G3: Examining underlying data. Having a snapshot of the underlying data is useful to assess the data's utility in downstream analysis. From Exemplars I and II, we determined this could be performed effectively via the **`assessment of univariate summary statistics` (T9)**. While more complex analyses could be carried out, it is challenging for one tool to anticipate and support these more complex analysis paths, but an initial univariate summary appeared to be useful starting point [67].

G4: Exporting Data to Analysis Tools. Finally, the goal of experimenting and trying different analytic data models depends on the ability to easily **`export` (T10)** the underlying data. Data should be exported to align with the analytic data model that experts have created.

Furthermore, our exemplars suggest a set of design requirements. The first requirement is **accessibility for analysts that vary in familiarity with the existing data model (R1)**. In our exemplars, we noted that level of familiarity related to the tasks that analysts performed. Those with greater familiarity, spent more time performing targeted browsing and look-up tasks. Those with less familiarity, spent more time browsing, but also reached out to other people for support. Another requirement was the ability to **support iterative refinement of data questions (R2)** as not all end-users have a clear goal at the outset of their investigation. Finally, the system should have sufficient capabilities to **limit transitioning between multiple tools (R3)**. This indicates that some data transformations between the existing and analytic data models should be handled within a common tool without having to constantly switch.

6 *Natto*

In this section, we describe the *Natto* prototype including both its visual and algorithmic design choices. Our design processes considered both the visual elements needed to support the creation of the analytic data models and the data transformations necessary for analysts to interact with the data. The *Natto* prototype is an interactive tool that enables an analyst to create an analytic data model from an existing data model.

6.1 Derivation of Intermediary Data

We derive three intermediary datasets that are used by *Natto* to ground the analyst's understanding in the existing data model and to help them create a new analytic data model. The data is a Consensus Tree, a Flattened Data Table, and a Data Graph. This XML document could be parsed by standard methods existing many of the analysis and database tools used by analysts (e.g, Tableau Prep, Python, R, or Javascript Libraries). However, analysts still needed to apply rather intensive transformative operations to create an analytic data model. We derive these intermediary data structures as a way to scaffold this process for supporting the design requirements (R1 - R3) articulated in the prior section. We describe their derivation in Algorithm 1 and show a simplified illustration of the data and this process in Fig. 2:

Algorithm 1 Pre-process (X_D)

Input: Directory of XML documents(X_D)
Output: Consensus Tree (C), Flattened Table (F), Data Graph (G)

```
(C,F) ← NULL
for x in X_D do
    (C',F') ← traveseXML(x)
    (C,F) ← getConsensus(C,C',F,F')
G ← dataGraph(C,F)
return(C,F,G)
```

Creating a Simplified Tree Structure. The first step constructs a Consensus Tree (C) derived by consolidating the trees within individual XML documents (C'). While prior research proposed more complex grammars and algorithmic approaches [2,4,9,44,46] to reconcile XML schemas, the schemas in our dataset were sufficiently similar despite small variations (see Section 3.1). We could hence take a simpler approach using a standard depth-first tree traversal. Our approach (`travseXML`) is similar to the one taken by Basic Inlining Technique described Shanmugasundaram *et al.* [49], which constructs both a graph and relational representation through a depth-first tree traversal. We remind the reader that a trivial tree traversal approach is possible because we leveraged existing tools for harmonizing across potential differences that may result from changes to the tree structure (Sect. 3.1).The `travseXML` essentially produces an intermediary data structure that stores a particular path in the tree and the data that it contains. The majority of data that our analysts cared about were stored in the leaf nodes of the trees because it contains more specific user generated content. The tags of the XML tree elements contained additional contextual data that were important for the software application, but were generally less interesting for analysis. The contextual data could be added in future interactions in a similar way that we add leaf node data.

The `getConsensus` algorithm traverses a single C' and determines whether a particular path in the XML tree has already been seen in C; if so, that path is not added to C as it is considered a redundant tree element; if the path is new, then it is added. This process of progressively adding new paths allow us to create consensus tree that helps analysts explore a simplified representation of the tree structure that addresses challenges of redundant tree elements (Section 3) and is intended to allow analysts to get an easier overview of the existing data model in order to explore it (`G1; T5`). In parallel to this traversal process, our algorithm identifies whether the tree node contains any data, which can be of type string, numeral, or Boolean. If the node does contain data, then its lineage is stored in a table along with the automatically detected data type and its value. This table is also updated over time to become a flat data file (F). These methods are not novel, but compared to prior work in data preparation and wrangling (see Section 2), which focused predominantly on tabular data, our approach affords a novel interaction to create an analytic data model.

Surfacing Internally Linked Nodes. To help analysts discover linkages between data nodes (`G2; T8`), we also derive a data graph. In Section 6.2.3, we describe how this graph is used to generate suggestions for the analyst; here we only detail the construction of this graph.

While graph and tree representations have been used in the past to visually link changes between XML trees [2, 22], our use of these data structures to both orient and motivate data transformation, is novel. We have defined a similarity metric to quantify the similarity of data containing nodes. The development of this metric is motivated by the goals, tasks, and requirements expressed in our exemplars:

$$S_{ij} = w_p * P_{ij} + w_d(D_{ij} * T_{ij}) \quad (1)$$

The similarity metric S_{ij} compares the proximity (P), data type (T), and shared data (D) between two nodes (i and j) in C. These values are summed in a weighted combination with the default values for weights w_p and w_c set to 0.3 and 0.7 respectively, assigning the greatest weight to connections derived from shared data. The final value of S is used to establish edges between two nodes in the XML tree, essentially adding links between the tree nodes; in fact, this graph+tree result inspired the system's name, as it reminded one of the authors of the sticky tendrils of the namesake Japanese snack.

Proximity is determined by the number of shared common ancestors; the more shared ancestors, the larger the value of P. We include proximity in the metric so that analysts can still consider useful nodes in their analytic data model without having to constantly consult the Consensus Tree. Next, the metric considers the shared data between two nodes. If these nodes have different data types (string, numeral, or Boolean), the right half of the equation becomes zero. If two nodes have the same data type, our algorithm computes a value for D and T. For integer and Boolean data, $D = 1$, since these values are not sufficiently specific to differentiate data similarity. For string data, D is the Jaccard similarity. We use the Jaccard similarity as it best allows us to detect the same string embedded within a larger string. For example, the string *'widgetA'* could also be easily identified with a comparative string that represents a calculation: *'sum[widgetA]/[totalWidget]'*. Finally, as the data type does appear to be important (e.g., an analyst wanting to find all text data), we included a weight to account for data type, T. We assign defaults values for T at 0.01, 0.5, and 1.0 for Boolean, numeric, and string data types respectively. Finally, these values are summed in a weighted combination with the default values for weight w_p and w_c set to 0.3 and 0.7 respectively, giving the greatest weight to connections derived from shared data. We prioritize shared string content in our similarity metric as we found that shared strings are less likely to result in spurious connections between tree nodes. The default values for w_p, w_c, and T were determined through iterative testing and discussion. We also enabled analysts to change these settings in the *Natto* interface.

The values of S are used to establish edges between internal and leaf nodes of the XML tree (Fig. 2). By default, $S > 0$ produces an edge, however, the *Natto* interface also allows analysts to set a new threshold for S and dynamically recompute the data graph's edges.

6.2 Visual and Interaction Design

Natto (Fig. 1) enables analysts to progressively construct an analytic data model. We describe the five components of this interface: the Tree Viewer, Model Canvas, Export, Suggestions, and Data View Panes. Once again, we link our design choices and alternatives we considered to exemplars goals (G#) and tasks (T#).

6.2.1 Tree Viewer Pane

The Tree Viewer Pane visualizes the Consensus Tree derived from the initial XML data source (see Section 6.1 for details). The analyst can interact with the tree in several ways to both understand and explore the structure of the existing data model (G1) and construct a new analytic data model (G2); examples of these interactions are shown in Fig. 1. The Consensus Tree representation removes redundant elements existing in the raw data and enables the analyst to engage more easily with the hierarchy and data within the XML data. We preserved this hierarchy as our discussion with analysts showed that at a high level, it matched the user's mental model, providing an effective overview (T1). The higher levels of the tree help analysts browse (T4) and look up (T2) elements of the leaf nodes. To browse the tree, an analyst can expand and contract the different nodes of the tree to identify (T3) relevant nodes for them to add to their analytic model. However, browsing the tree by this mechanism can also be slow and can impose a cognitive burden on the analyst to keep track of what parts of the tree they have explored or not. We address this limitation by enabling the analyst to perform a quick look-up (T2) and highlight selected elements of the tree to help orient them (T5).

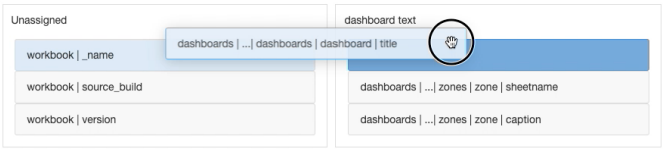

Fig. 3. Examples of two concept cards in the Canvas Pane, the default 'unassigned' and an analyst generated 'dashboard text' card. An analyst can drag nodes between concept cards.

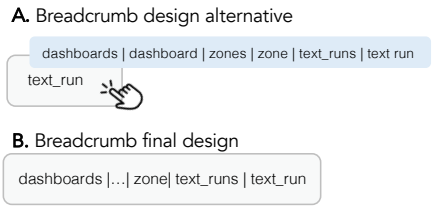

Fig. 4. a) An interaction-based breadcrumb design alternative that was considered, but ultimately discarded and b) the final breadcrumb design.

6.2.2 Model Canvas Pane

The Model Canvas Pane is intended to support analysts in their conceptualization of an analytic data model [14, 27, 63]. This pane enables an analyst to visually and interactively add and reorganize nodes in the original XML so that they can export a transformed data model for use in downstream analysis. Nodes can be added to the canvas from either the Tree Viewer or Suggestions Panes (T6); the analyst may select an individual leaf node or an entire subtree to add. Similarly elements can be removed by deselecting items in the Tree Viewer. Nodes added to the Canvas Pane are organized into 'concept cards' (Fig. 3), which follows from the user requirements of organizing elements of their analytic data model into semantic representations that are meaningful to them (G2; T7). Initially, there is only one concept card simply titled 'unassigned', but the analyst is able to add new concept cards on the fly, as shown in Fig. 1 and Fig. 3. When the analyst wishes to export their model (T10, G4) these concepts cards are used to define a set of tabular data outputs, transforming the initially XML data into a tabular format that is amenable to common data analysis tools.

In the initial versions of this prototype, only the node name was displayed in the canvas view, but we saw this representation as potentially disorienting given that names were repeated through the XML tree (by extension the Consensus Tree as well). This meant that the analyst could not always distinguish what they had placed onto the Canvas Pane. We introduced a 'breadcrumb' label to provide a sense of orientation (T5). Through an analysis of the Consensus Tree structure, we established that the breadcrumb label would be sufficiently unique if it contained a root node and a grandparent node (i.e., ancestral nodes that immediately proceeded the current node in the Consensus Tree) along with the node's own name. We use these breadcrumb labels for both the Canvas and Suggestion Panes.

In the Export Pane, the analyst can export their data into a single tabular data file with concepts card titles added as new categorical columns in the resulting table, or they may export multiple tables, with each table pertaining to one concept card (G4; T10).

6.2.3 Suggestions Pane

To further help the analyst discover elements they wish to add to their data model, *Natto* automatically generates suggestions in the Suggestions Pane. In Fig. 5 we show an example: the elements in gray reflect nodes in the Canvas Pane, each pertaining to text data (e.g., name, title)

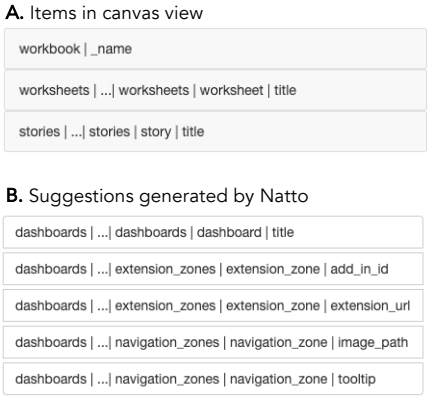

Fig. 5. Example suggestions generated by *Natto* in response to three nodes in the Canvas Pane. Here *Natto* suggests related fields in a part of a dashboard that the user has not yet added to the Canvas Pane.

Algorithm 2 Suggestion Algorithm (V,G)

Input: Snap Shot of the Canvas Pane (V), Data Graph (G)
Output: Ranked List of Field Suggestions (R),
1: (R,F) ← `NULL`
2: **for** v in V **do**
3: append `getNearestVertex`(v,G) to R
4: ($field$,$count$) ← `distinct`(R)
5: R ← `sort` field by count `in order` = $descending$
6: return(R)

in different subtrees of the Consensus Tree (e.g., worksheets, stories). Using these three items in the Model Canvas Pane, *Natto* provides suggestions, in ranked order, with up to five nodes in the Consensus Tree. The top-ranked node here is from the dashboards subtrees and also relates to title data. The objective of the Suggestions Pane is intended to support analysts that are both familiar and new to the existing data model by expediting their process. The Suggestions feature also supports those who are less familiar with the existing data model(G1).

In Algorithm 2, we show the processes of computing these suggestions on the fly. As the analyst creates their data model in the Canvas Pane (V), *Natto* periodically executes a task that takes a snapshot of V to generate suggestions. Each node in V contains a unique identifier that maps to the pre-computed data graph G (see Section 6.1 for details on how G is generated). For each node in V, *Natto* looks up its nearest neighbor in G and returns the result. The results are tabulated and rank ordered according to the number of nodes in V that share a common neighbor; the more overlap, the higher the rank of the suggestion. This ranked list is displayed in the Suggestions Pane and the user can add these nodes to the Canvas Pane simply by clicking on any of the suggestions. As we described in Section 6.1, some of the similarity computation that underlies the construction of G has elements that are modifiable by the user; the user can access these options via the Suggestions Pane (see Fig. 1).

6.2.4 Data Viewer Pane

Finally, the Data Viewer Pane visualizes a univariate summary of nodes in the Model Canvas Pane (Fig. 6). Not all nodes within the XML tree contain data that can be visualized; some nodes just add context to the hierarchical relationships of data that is stored. In general, leaf nodes of the XML tree contain data and can be helpful for analysts to get a glimpse of this data while conceptualizing their analytic data model(G3;T9). As already described earlier (see Section 6.1), there are different data types stored within the XML tree and these types are treated differently when being visualized. Numeric data is visualized as a histogram. Both string and Boolean data are visualized as bar charts. String data is treated as nominal data and as such can impose a high categorical cardinality. To address issues of cardinality, we limit the bar chart to contain at most the top 20 categories by the frequency of occurrence. Moreover, *Natto* automatically generates and displays subtitles that inform the analyst of the total number of categories actually shown in the visualization. In Fig. 6, we show an example of a visualization produced by *Natto* showing string data with 255 unique categories; the visualization shows the top 20 common categories (in this case, text containing the work 'state' is the most common) and creates a caption to indicate to the analyst what the total number of categories are.

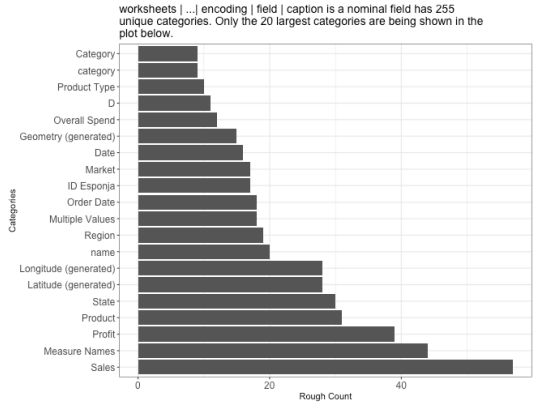

Fig. 6. Example of a univariate summary plot for a string (nominal) data corresponding to a leaf node in the Consensus Tree. *Natto* automatically generates a caption stating the total number of categories, displaying at most the top 20 categories by prevalence.

7 APPLICATION SCENARIOS

We present a preliminary demonstration of *Natto*'s capabilities based upon the exemplars described in Section 4. We first present a general pattern of usage that pertains to all the exemplars we described. We then dive deeper into Exemplars I and II, which resulted in specific analytic data models.

7.1 General Usage Scenarios

The analyst starts with the Tree Viewer Pane to examine the existing data model. They can browse the tree by expanding subtrees or searching for specific elements by name. They can select potentially interesting elements. This selection action will add the element to the Model Canvas Pane and into a generic unassigned card. The analyst can immediately view data associated with this element in the Data Viewer Pane to familiarize themselves with what is present in their dataset and make a judgement on whether the element is useful for their analysis or not. As the analyst includes more nodes, they can proceed to define new categories and move elements between concept cards. Through this process, they are defining a new analytic data model anchored in a conceptual understanding of the data that support their already formed (or evolving) data question. To support the analyst's discovery, the Suggestion Pane will also provide additional tree nodes based upon the content of the Model Canvas Pane. This feature is intended to support the discovery of new elements that the analyst had not yet considered, especially for those that are less familiar with the existing data model. To expedite the process, the analyst can choose to add these suggestions to their model.

7.2 Examining Text Content

Exemplar I (Sect. 4.1) sought to use text content to contextualize the end-users' visual analysis approaches. We now describe how they can achieve this goal using *Natto*. The analyst first begins by searching the tree for any obvious element names containing text that they could add to the Canvas Pane. The analyst begins to realise that text exists in multiple places throughout the XML Tree. For example, user-generated text within and alongside visualizations that end-users created from their data. Text within visualizations includes titles and axis labels, while text alongside visualizations includes captions. Deciding that

captions created by end-users are most interesting, the analyst limits their analysis to these text types. Wishing to further organize these findings, the analyst creates separate concept cards for text elements that are in titles, captions, and names as is shown in Fig. 1. They use suggestions to avoid having to expand and collapse various subtrees to find and select relevant elements. Satisfied with their selections, they export the data and bring it into another tool for further analysis. In Exemplar I, the analyst brought their analytic data model into Python to conduct a machine learning analysis.

7.3 Exploring Visual Strategies

The analyst has prior knowledge of the data model that anchors their search in the worksheet subtree in the Consensus Tree Pane. They search within the worksheet for data that indicates one worksheet is linked to another, as this indicates an attempt to link information across worksheets. Since worksheets contain visualizations constructed by end-users, these linkages between worksheets can be used to infer attempts by the end-user to construct linked views. To further support their search, the analyst can use the Data Viewer Pane to examine the data distribution elements describing the so-called 'action types'. These 'action' types indicate how the end-user is choosing to link worksheets. The analyst can make an assessment of the frequency and types of actions and identify those that might be more interesting. From this initially targeted examination, they can begin to expand their search, looking at the different encodings in the worksheet tree to get a sense of how they might link actions, encodings, and encoding types. Over time, their initial data model grows in complexity. Eventually, they export this data snapshot and begin to perform a more intensive descriptive statistical analysis; for the analyst in Exemplar II (4.2) this statistical analysis was carried out in Tableau.

8 DISCUSSION

Analysts routinely apply data transformations to iteratively explore multiple alternative representations of their data [28, 32, 37]. Often, such transformations are cast as data wrangling, but through discussions with four domain experts, we found that there is more depth to this process. Data often needs to be transformed from an existing data model (in our case an XML document) to an analytic data model that addresses a data question. One challenge of this transformation is determining the mapping between these data models, a problem that has been tackled extensively by the database community (see Section 2). Another challenge is the analysts' ability to ground themselves in an existing data model and envision a new one. Our research explored this latter challenge. We developed *Natto* as a demonstration of how analysts could be better supported in their pursuit of creating analytic data models, especially around hierarchical structures. While aspects of our findings are specific to the group of experts we interviewed and their datasets, other aspects can generalize to other applications. We briefly discuss the implications and limitations of our work.

8.1 Expanding our View of Data Preparation

The process of understanding and transforming data has always been seen as an integral part of data preparation [28]. However, it is our interpretation that, in practice, 'data understanding' is often prematurely limited to 'data quality understanding'. We encourage researchers to look beyond this limited scope and consider conceptual transformations of the data that arise when mapping from an existing model to a new analytic data model. In *Natto*, the Model Canvas Pane serves as the intermediary to facilitate the transformations between the existing tree data model and a tabular relational data model. We believe however, that there are more opportunities to consider for conceptual modeling in data preparation and within data wrangling specifically.

We also encourage researchers to look beyond tabular data preparation. Although tabular data remains the primary data source for many organizations [41], XML and other types of hierarchical data, represent a sizable and growing amount of data. Hierarchical data models present unique challenges. For example, in our study, the derivation of the right form of intermediary data to support exploration and transformation of a hierarchical data model, was especially important. While visualization research acknowledges the importance of derived data, it is rarely discussed as a design choice that researchers make alongside other visual and interaction design choices. We encourage researchers to discuss these choices as part of design studies as these decisions can generalize to other problems.

8.2 Incorporating Semantic Relationships

The most common data abstractions that research focuses on are inclusion (subtype-supertype, 'is a'), aggregation, and association (membership) relationships [56]. In addition to common data abstractions, research in linguistics, logic, and cognitive psychology has recognized the importance and prevalence of semantic relationships in data such as synonyms, inclusion, possession, and attribution [66].

Our implementation of *Natto* leverages some aspects of semantic relationships in the data through the consensus tree and by discovery relationships through the data graph. To some extent, we also enable users to define new semantic relationships in the data, ones that are more pertinent to their analysis goals, through the Model Canvas Pane. However, these semantic relationships play a more central role in helping the analysts construct their analytic data models. In our design study, the semantics of the XML data model reflected the visual analysis tool's architecture and, through a series of expert exemplars, we show how this existing data model is not amenable to analytic goals. This tension between the existing data semantics and whether they align with the analysts mental model, raises interesting research questions. Most notably, what happens when the transformations carried out by the user remove, alter, or invalidate, existing semantic relations in the existing data model? Balancing the trade-off between existing data semantics, which are typically established by an organization as well as personal data semantics, which constitute the analysts own notions about the data, offers interesting research directions to pursue.

8.3 Limitations

Our research was carried out in collaboration with a set of highly specialized experts and their data. This was both a strength and limitation of our findings and contributions. While we could probe their challenges of creating an analytic data model in depth, the *Natto* prototype is influenced by these experts' experiences. The methods that we chose to explore were also relatively simple, but we believe they were effective for the specific needs of our experts. However, choosing different methods can further improve upon the user experience. For instance, other methods can be used to explore content similarity among elements of the data graph or to generate recommendations. Finally, we chose visual components that are familiar and tend to be ubiquitous to the traversal and display of hierarchical data. However, richer visual and interaction encoding can be explored. Our primary goal in this analysis was to characterize the challenges of creating analytic data models and explore, as well as speculate upon, the algorithmic, visual, and interactions design choices that address this unmet need. While this may limit *Natto*'s ability to seamlessly generalize to other datasets, we nevertheless believe that aspects of *Natto* and our research findings will transfer to other application contexts in data preparation and wrangling.

9 CONCLUSION

We carried out a formative design study with domain experts posing different data questions on a common dataset. We identified their strategies and pain points for transforming their existing data model into a new analytic data model. We describe a set of goals, tasks, and design requirements for conceiving and creating an analytic data model. From this set, we have implemented the *Natto* prototype and a set of visual and interaction design choices. We provide application scenarios derived from our interviews with experts to demonstrate the utility of *Natto*. Our work links together important ideas in conceptual modeling, data transformation, and data wrangling. Finally, our research contributes to a broader understanding of how analysts organize and interrogate their data at the conceptual level. Our contributions bring together ideas from across multiple disciplines to present a novel visual and interactive experience for data analysts.

REFERENCES

[1] M. Aljubairah, S. Sampaio, H. A. Permana, and P. Sampaio. A conceptual approach to traffic data wrangling. In A. Calì, P. Wood, N. Martin, and A. Poulovassilis, eds., *Data Analytics*, pp. 9–22. Springer International Publishing, Cham, 2017.

[2] R. Amor, S. Bossung, J. Hosking, H. Stoeckle, and J. Grundy. Automated data mapping specification via schema heuristics and user interaction. In *Proceedings. 19th International Conference on Automated Software Engineering*, pp. 208–217. IEEE Computer Society, Los Alamitos, CA, USA, sep 2004. doi: 10.1109/ASE.2004.10035

[3] C. Arbesser, F. Spechtenhauser, T. Mühlbacher, and H. Piringer. Visplause: Visual data quality assessment of many time series using plausibility checks. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):641–650, 2017. doi: 10.1109/TVCG.2016.2598592

[4] G. J. Bex, F. Neven, and S. Vansummeren. Inferring XML schema definitions from XML data. In C. Koch, J. Gehrke, M. N. Garofalakis, D. Srivastava, K. Aberer, A. Deshpande, D. Florescu, C. Y. Chan, V. Ganti, C. Kanne, W. Klas, and E. J. Neuhold, eds., *Proceedings of the 33rd International Conference on Very Large Data Bases, University of Vienna, Austria, September 23-27, 2007*, pp. 998–1009. ACM, 2007.

[5] A. Bigelow, C. Nobre, M. Meyer, and A. Lex. Origraph: Interactive network wrangling. In *2019 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 81–92, 2019. doi: 10.1109/VAST47406.2019.8986909

[6] M. Blaschka, C. Sapia, and G. Höfling. On schema evolution in multidimensional databases. In M. Mohania and A. M. Tjoa, eds., *DataWarehousing and Knowledge Discovery*, pp. 153–164. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.

[7] M. Bloodgood and B. Strauss. Data cleaning for xml electronic dictionaries via statistical anomaly detection. In *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, pp. 79–86, 2016. doi: 10.1109/ICSC.2016.38

[8] F. Bodart, A. Patel, M. Sim, and R. Weber. Should optional properties be used in conceptual modelling? a theory and three empirical tests. *Information Systems Research*, 12, 12 2001. doi: 10.1287/isre.12.4.384.9702

[9] A. Boukottaya and C. Vanoirbeek. Schema matching for transforming structured documents. In *Proceedings of the 2005 ACM Symposium on Document Engineering*, DocEng '05, p. 101–110. Association for Computing Machinery, New York, NY, USA, 2005. doi: 10.1145/1096601.1096629

[10] D. Braga, A. Campi, and S. Ceri. Xqbe (xquery by example): A visual interface to the standard xml query language. *ACM Trans. Database Syst.*, 30(2):398–443, June 2005. doi: 10.1145/1071610.1071613

[11] D. Cashman, S. Xu, S. Das, F. Heimerl, C. Liu, S. R. Humayoun, M. Gleicher, A. Endert, and R. Chang. Cava: A visual analytics system for exploratory columnar data augmentation using knowledge graphs. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1731–1741, 2021. doi: 10.1109/TVCG.2020.3030443

[12] S. Ceri, S. Comai, E. Damiani, P. Fraternali, S. Paraboschi, and L. Tanca. Xml-gl: A graphical language for querying and restructuring xml documents. *Comput. Netw.*, 31(11–16):1171–1187, May 1999. doi: 10.1016/S1389-1286(99)00014-6

[13] B. Chan, L. Wu, J. Talbot, M. Cammarano, and P. Hanrahan. Vispedia: Interactive visual exploration of wikipedia data via search-based integration. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1213–1220, 2008. doi: 10.1109/TVCG.2008.178

[14] P. P.-S. Chen. The entity-relationship model—toward a unified view of data. *ACM Trans. Database Syst.*, 1(1):9–36, Mar. 1976. doi: 10.1145/320434.320440

[15] D. Cohn and R. Hull. Business artifacts: A data-centric approach to modeling business operations and processes. *IEEE Data Eng. Bull.*, 32(3):3–9, 2009.

[16] A. Crisan, B. Fiore-Gartland, and M. Tory. Passing the data baton : A retrospective analysis on data science work and workers. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1860–1870, 2021. doi: 10.1109/TVCG.2020.3030340

[17] N. Dedić and C. Stanier. Towards differentiating business intelligence, big data, data analytics and knowledge discovery. In F. Piazolo, V. Geist, L. Brehm, and R. Schmidt, eds., *Innovations in Enterprise Information Systems Management and Engineering*, pp. 114–122. Springer International Publishing, Cham, 2017.

[18] M. Dudáš, S. Lohmann, V. Svátek, and D. Pavlov. Ontology visualization methods and tools: a survey of the state of the art. *The Knowledge Engineering Review*, 33:e10, 2018. doi: 10.1017/S0269888918000073

[19] R. Elmasri and S. B. Navathe. *Fundamentals of Database Systems*. Pearson, 7th ed., 2015.

[20] M. Erdmann and R. Studer. How to structure and access xml documents with ontologies. *Data & Knowledge Engineering*, 36:317–335, 03 2001. doi: 10.1016/S0169-023X(00)00048-3

[21] T. Furche, G. Gottlob, L. Libkin, G. Orsi, and N. Paton. Data wrangling for big data: Challenges and opportunities. In *Advances in Database Technology — EDBT 2016*, Advances in Database Technology, pp. 473–478. University of Konstanz, 2016. 19th International Conference on Extending Database Technology, EDBT 2016 ; Conference date: 15-03-2016 Through 18-03-2016. doi: 10.5441/002/edbt.2016.44

[22] M. Graham and J. Kennedy. A survey of multiple tree visualisation. *Information Visualization*, 9(4):235–252, 2010. doi: 10.1057/ivs.2009.29

[23] 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 *Proceedings of the 14th International Conference on Knowledge Technologies and Data-Driven Business*, i-KNOW '14. Association for Computing Machinery, New York, NY, USA, 2014. doi: 10.1145/2637748.2638423

[24] P. Hauer. Why relational databases are not the cure-all. strength and weaknesses. `https://phauer.com/2015/relational-databases-strength-weaknesses-mongodb/`. Accessed: 2021-05-27.

[25] J. Heer and A. Perer. Orion: A system for modeling, transformation and visualization of multidimensional heterogeneous networks. In *2011 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 51–60, 2011. doi: 10.1109/VAST.2011.6102441

[26] M. Jabbari and J. Recker. Combined use of conceptual models in practice: An exploratory study. *Journal of Database Management*, 28:56–88, 06 2017. doi: 10.4018/JDM.2017040103

[27] Z. Jin, M. R. Anderson, M. Cafarella, and H. V. Jagadish. Foofah: Transforming data by example. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, p. 683–698. Association for Computing Machinery, New York, NY, USA, 2017. doi: 10.1145/3035918.3064034

[28] 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):271–288, 2011. doi: 10.1177/1473871611415994

[29] S. Kandel, A. Paepcke, J. Hellerstein, and J. Heer. Wrangler: Interactive visual specification of data transformation scripts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, p. 3363–3372. Association for Computing Machinery, New York, NY, USA, 2011. doi: 10.1145/1978942.1979444

[30] S. Kandel, A. Paepcke, J. Hellerstein, and J. Heer. Enterprise data analysis and visualization: An interview study. *Visualization and Computer Graphics, IEEE Transactions on*, 18:2917–2926, 12 2012. doi: 10.1109/TVCG.2012.219

[31] S. Kasica, C. Berret, and T. Munzner. Table scraps: An actionable framework for multi-table data wrangling from an artifact study of computational journalism. *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–1, 2020. doi: 10.1109/TVCG.2020.3030462

[32] G. Klein, J. Phillips, E. Rall, and D. Peluso. A data-frame theory of sensemaking. *Expertise out of Context: Proceedings of the Sixth International Conference on Naturalistic Decision Making*, pp. 113–155, 01 2007.

[33] M. Klein, D. Fensel, F. V. Harmelen, and I. Horrocks. The relation between ontologies and xml schemas. In *Linkoping Electronic Articles in Computer and Information Science*, vol. 6, 2001.

[34] M. Koehler, A. Bogatu, C. Civili, N. Konstantinou, E. Abel, A. A. A. Fernandes, J. Keane, L. Libkin, and N. W. Paton. Data context informed data wrangling. In *2017 IEEE International Conference on Big Data (Big Data)*, pp. 956–963, 2017. doi: 10.1109/BigData.2017.8258015

[35] H. Lam, M. Tory, and T. Munzner. Bridging from goals to tasks with design study analysis reports. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):435–445, 2018. doi: 10.1109/TVCG.2017.2744319

[36] P. Leinonen. Automating xml document structure transformations. In *Proceedings of the 2003 ACM Symposium on Document Engineering*, DocEng '03, p. 26–28. Association for Computing Machinery, New York,

NY, USA, 2003. doi: 10.1145/958220.958226

[37] J. Liu, N. Boukhelifa, and J. R. Eagan. Understanding the role of alternatives in data analysis practices. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):66–76, 2020. doi: 10.1109/TVCG.2019.2934593

[38] S. Liu, G. Andrienko, Y. Wu, N. Cao, L. Jiang, C. Shi, Y.-S. Wang, and S. Hong. Steering data quality with visual analytics: The complexity challenge. *Visual Informatics*, 2(4):191 – 197, 2018. doi: 10.1016/j.visinf.2018.12.001

[39] Z. Liu, S. B. Navathe, and J. T. Stasko. Network-based visual analysis of tabular data. In *2011 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 41–50, 2011. doi: 10.1109/VAST.2011.6102440

[40] P. Mayadewi, B. Sitohang, and F. N. Azizah. Scheme mapping for relational database transformation to ontology: A survey. In *2017 International Conference on Data and Software Engineering (ICoDSE)*, pp. 1–6, 2017. doi: 10.1109/ICODSE.2017.8285866

[41] A. M. P. Milani, F. V. Paulovich, and I. H. Manssour. Visualization in the preprocessing phase: Getting insights from enterprise professionals. *Information Visualization*, 19(4):273–287, 2020. doi: 10.1177/1473871619896101

[42] M. Necaský. Reverse engineering of XML schemas to conceptual diagrams. In M. Kirchberg and S. Link, eds., *Conceptual Modelling 2009, Sixth Asia-Pacific Conference on Conceptual Modelling (APCCM 2009), Wellington, New Zealand, January 20-23 2009*, vol. 96 of *CRPIT*, pp. 117–128. Australian Computer Society, 2009.

[43] A. Qtaish and K. Ahmad. Xancestor: An efficient mapping approach for storing and querying xml documents in relational database using path-based technique. *Knowledge-Based Systems*, 114:167 – 192, 2016. doi: 10.1016/j.knosys.2016.10.009

[44] A. Raffio, D. Braga, S. Ceri, P. Papotti, and M. A. Hernandez. Clip: a visual language for explicit schema mappings. In *2008 IEEE 24th International Conference on Data Engineering*, pp. 30–39, 2008. doi: 10.1109/ICDE.2008.4497411

[45] V. Raman and J. M. Hellerstein. Potter's wheel: An interactive data cleaning system. In *Proceedings of the 27th International Conference on Very Large Data Bases*, VLDB '01, p. 381–390. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001.

[46] G. G. Robertson, M. P. Czerwinski, and J. E. Churchill. Visualization of mappings between schemas. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, p. 431–439. Association for Computing Machinery, New York, NY, USA, 2005. doi: 10.1145/1054972.1055032

[47] N. Sambasivan, S. Kapania, H. Highfill, D. Akrong, P. Paritosh, and L. M. Aroyo. "everyone wants to do the model work, not the data work": Data cascades in high-stakes ai. In *CHI'21*, CHI '21. Association for Computing Machinery, New York, NY, USA, 2021. doi: 10.1145/3411764.3445518

[48] V. Shah and A. Kumar. The ml data prep zoo: Towards semi-automatic data preparation for ml. In *Proceedings of the 3rd International Workshop on Data Management for End-to-End Machine Learning*, DEEM'19. Association for Computing Machinery, New York, NY, USA, 2019. doi: 10.1145/3329486.3329499

[49] J. Shanmugasundaram, K. Tufte, C. Zhang, G. He, D. J. DeWitt, and J. F. Naughton. Relational databases for querying xml documents: Limitations and opportunities. In *Proceedings of the 25th International Conference on Very Large Data Bases*, VLDB '99, p. 302–314. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[50] D.-H. Shin and K.-H. Lee. Towards the faster transformation of xml documents. *Journal of Information Science*, 32(3):261–276, 2006. doi: 10.1177/0165551506064391

[51] H. Shiu and J. Fong. Reverse engineering from an XML document into an extended DTD graph. *J. Database Manag.*, 20(2):38–57, 2009. doi: 10.4018/jdm.2009040103

[52] I. C. S. Silva, G. Santucci, and C. M. D. S. Freitas. Visualization and analysis of schema and instances of ontologies for improving user tasks and knowledge discovery. *Journal of Computer Languages*, 51:28 – 47, 2019. doi: 10.1016/j.cola.2019.01.004

[53] G. Simsion. *Data Modeling Theory and Practice*. Technics Publications, LLC, Denville, NJ, USA, 2007.

[54] M. K. Singh, D. K. G., and D. Taniar, eds. *Effective Big Data Management and Opportunities for Implementation:*. Advances in Data Mining and Database Management. IGI Global, 2016. doi: 10.4018/978-1-5225-0182-4

[55] A. Srinivasan, H. Park, A. Endert, and R. C. Basole. Graphiti: Interactive specification of attribute-based edges for network modeling and visualization. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):226–235, 2018. doi: 10.1109/TVCG.2017.2744843

[56] V. C. Storey. Understanding semantic relationships. *The VLDB Journal*, 2(4):455–488, Oct. 1993. doi: 10.1007/BF01263048

[57] I. G. Terrizzano, P. M. Schwarz, M. Roth, and J. E. Colino. Data wrangling: The challenging yourney from the wild to the lake. In *CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*. www.cidrdb.org, 2015.

[58] H. Topi and V. Ramesh. Human factors research on data modeling: A review of prior research, an extended framework and future research directions. *J. Database Manag.*, 13(2):3–19, 2002. doi: 10.4018/jdm.2002040101

[59] W. van der Aalst. *Data Science in Action*, pp. 3–23. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016. doi: 10.1007/978-3-662-49851-4_1

[60] I. Varlamis and M. Vazirgiannis. Bridging xml-schema and relational databases: A system for generating and manipulating relational databases using valid xml documents. In *Proceedings of the 2001 ACM Symposium on Document Engineering*, DocEng '01, p. 105–114. Association for Computing Machinery, New York, NY, USA, 2001. doi: 10.1145/502187.502203

[61] B. M. von Zernichow and D. Roman. Usability of visual data profiling in data cleaning and transformation. In *OTM Conferences*, 2017.

[62] Y. Wand, V. C. Storey, and R. Weber. An ontological analysis of the relationship construct in conceptual modeling. *ACM Trans. Database Syst.*, 24(4):494–528, Dec. 1999. doi: 10.1145/331983.331989

[63] Y. Wand and R. Weber. Research commentary: information systems and conceptual modeling a research agenda. *Information Systems Research*, 13:363–, 01 2002.

[64] Wes McKinney. Data structures for statistical computing in python. In Stéfan van der Walt and Jarrod Millman, eds., *Proceedings of the 9th Python in Science Conference*, pp. 56 – 61, 2010. doi: 10.25080/Majora-92bf1922-00a

[65] H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016.

[66] M. E. Winston, R. Chaffin, and D. Herrmann. A taxonomy of part-whole relations. *Cognitive Science*, 11(4):417–444, 1987. doi: 10.1016/S0364-0213(87)80015-0

[67] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2016.