Universität Rostock Traditio et Innovatio

Dissertation

A multifaceted Model for the layered Orchestration of independent Visual Analysis Tools

zur

Erlangung des akademischen Grades

Doktor-Ingenieur (Dr.-Ing)

der Fakultät für Informatik und Elektrotechnik

der Universität Rostock

vorgelegt von Lars Nonnemann

geboren am 12.06.1994 in Crivitz

wohnhaft in Ziesendorf Ortsteil Fahrenholz

Rostock, 11.07.2025

https://doi.org/10.18453/rosdok_id00005049

Erstgutachter: Prof. Dr.-Ing. Uwe von Lukas	Universität Rostock
Zweitgutachter: Associate Prof. Dr.-Ing. Hans-Jörg Schulz	Universität Aarhus
Drittgutachter: Prof. Dr. techn. habil. Stefan Bruckner	Universität Rostock

Eingereicht am: 11.11.2024

Verteidigt am: 30.06.2025

Schlagwörter / Keywords:

Visual Analytics, Information Visualization, Workflow Visualization, Tool Coordination, Data Integration, Data Analytics, Data Exchange

Klassifikation / Classification:

Visualization → Visual Analytics → User-oriented Toolchaining

Kurzfassung

Die wachsende Menge an Informationen, die wir täglich erzeugen und sammeln, erfordert neue, ausgeklügelte Lösungen, um Analysen zu erleichtern und Erkenntnisse daraus zu gewinnen. Aus diesem Grund werden stetig neue "Visual Analytics Werkzeugeëntwickelt, die auf spezifische Anwendungsdomänen zugeschnitten sind. Die Umsetzungen reichen dabei von deskriptiven Bibliotheken bis hin zu vollumfänglichen Frameworks, die darauf abzielen, den Arbeitsablauf von Domänenexperten zu erleichtern. Die heterogene und isolierte Natur dieser Tools stellt jedoch eine Herausforderung dar, wenn unterschiedliche Datenquellen und Funktionalitäten kombiniert werden müssen. Um mit diesem Problem umzugehen, beinhalten aktuelle Methoden zur Koordination von Tools entweder die Erstellung eines neuen Systems, das alle Funktionen integriert, oder erfordern, dass Benutzer in den verschiedenen Phasen einer Analyse unterschiedliche Tools manuell anwenden. Beide Ansätze sind ressourcenintensiv und umständlich, was die Benutzerfreundlichkeit und Effizienz für Domänenexperten einschränkt.

Um eine flexiblere Lösung zu finden, präsentiert diese Arbeit einen neuartigen Ansatz zur Orchestrierung von Visual Analytics Werkzeugen durch ein mehrschichtiges, workflow-orientiertes Koordinationsmodell. Dieses Modell zielt darauf ab, die konzeptionellen, räumlichen und zeitlichen Trennungen der interaktiven visuellen Analyse zu überwinden, welche die effektive Anwendung von Multi-Tool-Workflows derzeit behindern. Durch ein leichtgewichtiges Koordinationsmodell ermöglicht der Ansatz eine flexible, minimal invasive Kopplung auf der Datenebene zwischen ansonsten unabhängigen Werkzeugen. Ergänzt durch einen ausgearbeiteten Ansatz für UI-Ensembles auf der Ansichtsebene bietet diese Arbeit daher ein praktikable Lösung für das Mantra İntegrate what is necessary and couple what is possible", um den domänenspezifischen Anforderungen der Benutzer gerecht zu werden.

Das vielschichtige Modell wurde durch eine Konfigurations- sowie eine Steuerungsschnittstelle operationalisiert und als Open-Source-Anwendung implementiert. Die Anwendung wurde dabei in einem dreistufigen Evaluierungsprozess gemeinsam mit verschiedenen Benutzergruppen entwickelt und optimiert, um die benutzerorientierte Erstellung und Ausführung von Workflows in Form von analytischen Werkzeugketten zu unterstützen. Die schrittweise Evaluierung bestätigte die Umsetzbarkeit des Ansatzes in praktischen Szenarien, insbesondere in der medizinischen Datenanalyse, wo die Effizienz und Benutzerfreundlichkeit der Arbeitsabläufe durch eine Anpassung an die realen Analyseprozesse der Benutzer verbessert wurden.

Durch die Verknüpfung unabhängiger visueller Analysen über eine einheitliche Schnittstelle legt diese Arbeit den Grundstein für nachhaltige, anpassungsfähige Prozesse mit mehreren Visual Analytics Werkzeugen und fördert den Übergang zu zugänglicheren, benutzerorientierten und integrierten Lösungen im Bereich der visuellen Analyse.

Abstract

The growing amount of information that is generated and collected every day necessitates new sophisticated solutions to facilitate analysis and derive insights. Hence, new domain-specific visual analytics tools are constantly being developed, ranging from descriptive libraries to fully defined frameworks meant to simplify the user's workflow. However, the heterogeneous and isolated nature of these tools creates challenges for scenarios in which diverse data sources and functionalities need to be combined. Current methods that deal with tool coordination either involve creating a new system that integrates all functions or require users to utilize different tools at various stages of the analysis manually. In consequence, both approaches are resource-intensive and cumbersome, limiting usability and efficiency for domain experts. In order to find a more flexible solution, this thesis proposes a novel approach to the orchestration of visual analytics tools through a layered, flow-oriented coordination model. This model aims to overcome the conceptual, spatial, and temporal separations of interactive visual analysis that impede effective multi-tool workflows. Using a lightweight coordination model, it emphasizes flexible, minimally invasive coupling between otherwise independent visual analytics tools on the data. Together with an elaborated approach for user interface ensembles on the view level, it thereby presents itself as a viable solution to integrate what is necessary and couple what is possible according to the user's domain-specific requirements. This multifaceted model is operationalized through both a configuration and a control interface implemented as an open-source software application. It was designed and re-shaped through a three-stage evaluation process together with various user groups to aid the editorial creation and execution of user-defined workflows in the form of analytical toolchains. The stage-wise evaluation thereby demonstrated the model's effectiveness in practical application scenarios, specifically in medical data analysis, where it improved workflow efficiency and usability by aligning tools with users' real-world analytical processes. By bridging independent visual analytics through a unified interface, this work lays the groundwork for sustainable, adaptable multi-tool processes, advancing the field of visual analytics toward more accessible, user-oriented, and workflow-integrated solutions.

Contents

1 Introduction

1.1 Motivation

The analysis of data with visual representations of real-world phenomena is a widespread, interdisciplinary subject in computer science commonly known as *Visual Analytics*. Within this subject, various data analytics and information visualization methods have been established by providing specific tools that aim to support the user in their domain-dependent task. Each of those *visual analytics tools* provides different procedures and functionalities to detect or solve certain use case problems of a domain. As a result, a wide variety of visual analytics tools have emerged over the past decades, ranging from simple statistical plot diagrams to comprehensive visualization frameworks offering a wide variety of alternating representations.

This diversity is quite understandable, given the vast amount of information that we generate and collect every day. In the past, only a few data values of a machine had to be analyzed. Today multi-factorial data streams pose a challenge for various commercial and scientific application areas. The resulting technologies increase complexity and demand additional effort when creating domain-specific solutions, as each new domain quest inquiry necessitates the development of a unique tool or framework. While years ago, domain experts would have resided with a proper visualization of their domain, they now request broad functionality for visualization, interaction, and computational analysis – e.g., filtering large data sets, inspecting details of a visualization, interacting with the shown, and potentially documenting or sharing findings in multiple ways. This not only requires significant resources and effort for creation, adaptation, and refinement, but further confines algorithms and techniques to highly domain-specific tools, despite their potential usefulness in other contexts. Additionally, new visual analytics tools are becoming increasingly overloaded with different types of information. To compensate for this distress, visualization authors aim to either reduce the visual load of an application or split up its overall content in separate views. While topics like user-guidance and reduction of visual clutter are already considered for individual visual analytics tools, their use in domain workflows is for the most part unspecified. However, this is simultaneously the most effective point to add value towards a sustainable development of of visualization solutions. Over time, two dedicated philosophies for dealing with multiple heterogeneous systems have therefore emerged: 1) the combined use of multiple visual analytics tools through pairwise coupling and 2) the integration of existing functions into monolithic expert systems.

A common practice for handling the problem of disconnected functionality is to deploy multiple visual analytics tools, one after the other, to solve tasks independently.

Typically, this approach involves manually launching a visual analytics tool to import data, identifying the required information, and passing refined data or gained insights to the next visual analytics tool based on the user's predefined workflow. These activities not only require compatible data exchange between the individual visual analytics tools due to different encoding and standards, but also constitute an unnecessary workload for the user, as important time is lost through rudimentary import and export tasks. Additionally, switching user interfaces might increase the workload for users due to necessary reorientation, which potentially leads to incorrect results due to a lack of attention in between view transitions.

Another solution regarding the problem of disconnected functionality is the ongoing integration of existing visual analytics tools in rather comprehensive software products. Many of the commercially successful products are stand-alone applications that attempt to cover all functionality of previous generations, resulting in extensive user interfaces that are hardly comprehensible for the average user. These monolithic analysis systems usually provide a lot of hidden features, while most of the time, only certain parts of it are actually needed for the analysis task at hand. The development of such feature-rich applications requires not only code-level access to all the involved visual analytics tool functions, but also enormous development costs in employee labor and time that can only be met by appropriate production teams.

1.2 Problem Description

Instead of regularly developing new visual analytics tools for specific tasks or enhancing existing monolithic systems for every possible usage scenario, it is necessary that visualization authors aim for the integration of functionality into the user's domain workflow. While the required visual analytics tools already exist for many use cases, their combined use is obstructed by the heterogeneity and the manual bindings in specific software solutions. Domain users, on the other hand, often lack the time or technical expertise to manually coordinate the various visual analytics tools according to their common workflows. Neither the utilization of individual visual analytics tools through pairwise coupling nor the integration of visual analytics tools in a monolithic system are helpful to this cause, as none of the mentioned philosophies supports the domain user in the long run. The usability for each of the two approaches is strongly related to the expected technical knowledge of the user, the required resources and the given application scenario. To make better use of existing visual analytics tools, the flow-oriented nature of user workflows should be carried out by integrating what is necessary and coupling what is possible to achieve an interlinked perspective for next-generation visual analytics tools.

1.3 Research Questions and Goal

The existing research in this area shows that many questions relevant to this topic have either not been answered yet mainly been looked at in isolation. One should cover

the following research questions to gain insight into visual analytics tool inter-process communication.

- Which principal types of data exchange between visual analytics tools exist?
- What parts of the data can or should be exchanged between visual analytics tools?
- How does someone annotate the data flow to provide meaningful information?
- What should be shown to represent the user's workflow?
- How can the workflow be represented in a meaningful way?
- How can the user interactively control the UI ensembles?

Therefore, the overarching goal of an implementation in this regard should be to provide a unified user interface that answers these questions to facilitate the exchange of data between different tools on the data level and arrange the visual output generated by different tools on the view level. As stated by Streit et al. [Str+12], such a unified representation can be envisioned as a heterogeneous information landscape in which information foraging and sense-making take place.

1.4 Contribution

This thesis aims to provide methods for utilizing and re-purposing existing visual analytics tools in a way that is both domain-independent and applicable for general use. To this cause, flow-oriented behavior between multiple visual analytics tools was investigated to establish a loose coupling between them into specialized ensembles as needed. This type of coupling among independent tools is declared as *lightweight coordination* [Sch+19b], as it is minimally invasive, pairwise, and opportunistic in utilizing whichever interface a visual analytics tool offers. Following the concepts of nested visualization design [Mun15] and directness in interactive visual data analysis [TS20], the lightweight nature is further enhanced by a layered attribution. The resulting model for the layered coordination is designed and implemented for managing flow-oriented processes among visual analytics tools by:

- Solving the *temporal separation* through capturing the intended order in which tools are used in the *usage flow*, so as to know between which tools coordination is necessary and only to realize it between them,
- Solving the *conceptual separation* through capturing different characteristics of data exchange between each pair of visual analytics tools in the *data flow*, so as to specify individually how the data exchange is performed,
- Solving the *spacial separation* through capturing the actions of the domain expert and the system in the *control flow*, so as to react and signal necessary adjustments in the visualization.

The previously mentioned research questions are analyzed following this flow-oriented structure and answered on two levels: 1) on the data level to facilitate the exchange of data between different tools, and 2) on the view level to arrange the visual output generated by different tools.

Based on the investigation and the resulting models, a suitable setup of the multiple user interfaces (UIs) was developed. This setup is meant to be seen as a unified interface for visual analytics, since it supports the combination of visual analytics tools and allows access to data and functions of all tools to be used within a given domain workflow. It therefore serves data-flow-oriented mechanisms for unified access onto otherwise separately used individual visual analytics tools, rather than developing a general-purpose application that permanently integrates stand-alone tools into a fully-fledged software solution. This concept enables users with limited or non-existent technical experience to integrate the functionalities of existing visual analytics tools into their workflow with minimal or, ideally, no implementation-related adjustments.

The proposed model was implemented as an open-source software solution called the Analytical Process Configurator (AnyProc). For evaluation purposes, the methodologies of this thesis have been demonstrated for three specific medical data analysis scenarios showcasing the proposed design process, coordination model, and UI setups in technique-, workflow-, and application-centered examples.

1.5 Outline

Following this introductory Chapter 1, Chapter 2 provides the necessary background for understanding the broad research area of visual analytics and its challenges, as well as the related work on the topic of independent tool combination and coordination. To this cause, Section 2.1 of the chapter provides necessary definitions and establishes foundational terminology essential for exploring the diverse aspects that are important for data exchange between visual analytics tools. Subsequently, Section 2.2 of the same chapter provides an overview of the existing solutions based on a systematical investigation on the state of the art. This includes visual analytics coordination approaches in terms of data management, analytics, and visualization, as each of these aspects is critical for facilitating flow-oriented visual analysis across independent tools.

Based on this, Chapter 3 identifies and examines the primary challenges for the separation of concern involved in the orchestration of such independent visual analytics tools. These challenges are used to engage with the previously mentioned research questions from Section 1.3 to build hypotheses for potential solutions and explore limitations. Overall, the chapter builds and refines the requirements and limitations to define the goal of this thesis in form of a new mantra to “integrate what is necessary and couple what is possible”.

With the prerequisites aside, Chapter 4 presents the conceptual core of this thesis by providing the multifaceted model for the orchestration of layered separation requirements as the central contribution based on the previously achieved publications.

Starting with identifying relevant entries and establishing minimal invasive entry

points for cross-tool coordination, a lightweight model is presented to establish a pairwise connection between visual analytics tools. This idea is based on the first collaborative publication from 2019, titled as lightweight coordination of multiple independent visual analytics tools [Sch+19b], which was published in the proceedings of the International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP) (doi: 10.5220/0007571101060117). The concept was slightly altered and extended in 2020, titled as a layered approach to lightweight toolchaining in visual analytics [Sch+20], to be included in the Springer chapter for selected papers on Computer Vision, Imaging and Computer Graphics Theory and Applications. In contrast to the original publication, this thesis focuses more on the first part of the defined mantra to "integrate what is necessary" in the pairwise coordination, before moving on towards more complicated structures such as the coordination graph.

Following up, the data exchange between pairwise coupled visual analytics tools is examined to find characteristics for interoperability that can be applied to various independent visual analysis systems. This idea is based on a publication from 2020, titled as a characterization of data exchange between visual analytics tools [Non+20], which was published in the proceedings of the IEEE International Conference on Information Visualization (IV) (doi: 10.1109/IV51561.2020.00066). The resulting model for the design and evaluation of visual analytics tool ensembles represents thereby a fundamental contribution by providing the "space of possible" that is needed to define the second part of the mantra to "couple what is possible".

Moving up from pairwise coupling, the toolchains and their underlying structure of the coordination graph are further defined as the next step towards orchestrated visual analytics tool coordination on the data and view levels. This idea is based on a publication from 2022, titled a data-driven platform for the coordination of independent visual analytics tools [Non+22], which was published as an invited article in the Elsevier Computers and Graphics journal (doi: 10.1016/j.cag.2022.05.023). In contrast to the original publication, this thesis follows the separation of concerns discussed in Chapter 3 to enhance the simplified model and deal with the conceptual separation through layered coordination, spacial separation through the structured use of user interface ensembles and temporal separation through an interactive editor for the configuration of toolchains.

Chapter 5 then goes on to explain how the described conceptual model can be used to implement a unified interface for the orchestration of visual analytics tools. It thereby explains the overarching structure of the Analytical Process Configurator (AnyProc) as it has been used for demonstration purposes in previous publications [Non+21; Non+22]. This is enhanced by examining other research efforts conducted during the UnIVA project [Non+19; Sch+21; Röh+23] to showcase how the data-driven model can be used as an entry point towards a holistic view on visual analytics tool coordination.

The provided implementation efforts are then evaluated in Chapter 6 as part of a three-stage procedure including technique-centered, workflow-centered, and application-centered strategies, which are each based on different scenarios and user groups. The results of the technique-centered evaluation are thereby closely bound to the publication from 2019, titled Health@Hand A Visual Interface for eHealth Monitoring [Non+19], which was published at the IEEE symposium on computers and communications (ISCC)

(doi: 10.1109/ISCC47284.2019.8969647). The same applies to the results of the workflow-centered evaluation that are closely bound to the publication from 2021, titled as a customizable coordination of independent visual analytics tools [Non+21], which was published at the Eurographics international workshop on Visual Analytics (EuroVA) (doi: 10.2312/eurova.20211094). This also applies to the results of the application-centered evaluation that are closely bound to the publication from 2023, titled towards a unified user interface for visual analysis of retinal data in ophthalmology [Röh+23], which was published as an arXiv article (doi: 10.48550/arXiv.2301.01840).

Chapter 7 follows up on the results of the evaluation to compare them to the previously defined research questions and hypothesis from Chapter 3 to either approve or falsify the assumptions of this thesis.

Finally, Chapter 8 summarizes the results and provides possible directions for future work towards the spread into various application fields, enhancements for data connectivity, delayed data transfers, and intelligent guidance approaches.

For convenience, in the upcoming chapters, the basic definition of Visual Analytics Tools will be referred to by the acronym VAT from this point on.

1.6 Designation of Joint Work

The methods applied in this thesis, were developed in close collaboration with Dr. Martin Röhlig supervised by Prof. Dr.-Ing. habil. Heidrun Schumann in context of the DFG project UnIVA. While their work was primarily focused on the visual layout and representation of visual analytics tools, this thesis is set to be focused on the combinatory aspects of data exchange. Hence, this thesis contains part of the general concepts that are published in previous research and corresponding papers as described in Section 1.5. Fundamental ideas of our joint work have been extended in a more abstract way to accommodate the different aspects that were developed in the scope of this thesis and go beyond the original publications. The presented use cases showing the application originate from our joint work and are discussed accordingly.

2 Background on Visual Analytics Tool Coordination

This chapter focuses on the basic principles and existing research necessary to understand how the original research questions can be addressed. It contains descriptions of the fundamental visual analytics research as well as the current state of the art in relation to existing approaches for coupling and coordinating such visual analytics applications.

For this purpose, the differences between data analysis and information visualization are explicitly described to align the vision of visual analytics and define the scope for VATs regarding their interdisciplinary nature. This includes traditional problems, techniques, and mantras that are necessary to understand the topics and concepts presented in this work.

Subsequently, existing approaches are showcased with various examples to discuss problems related to data management, analysis, and visualization. This is meant to motivate the properties of VATs and showcase the historical development from disparate research fields towards the multitude of available VATs that are core to the research questions of this thesis.

2.1 Fundamentals of Visual Analytics

The following section serves to categorize the research field of visual analytics. It discusses its historical development, common definitions, and areas of application.

2.1.1 The Information Overload Problem

In 2008, Keim et al. [Kei+08] described an important motivator for the further development of visual analytics known as the *information overload problem*:

> "We are living in a world which faces a rapidly increasing amount of data to be dealt with on a daily basis. In the last decade, the steady improvement of data storage devices and means to create and collect data along the way influenced our way of dealing with information: Most of the time, data is stored without filtering and refinement for later use. Virtually every branch of industry or business and any political or personal activity nowadays generate vast amounts of data. Making matters worse, the possibilities to collect and store data increase at a faster rate than our ability to use it for making decisions. [...] Due to information overload, time and money are wasted,

> scientific and industrial opportunities are lost because we still lack the ability to deal with the enormous data volumes properly."

To put it more directly, the problem of information overload shows that there is a risk of getting lost in data that is unrelated to the current task, processed inadequately, or presented in an unsuitable way.

For this reason, numerous tools have been developed in recent decades to visually prepare and analyze data across a wide range of application domains. This led to a turning point in which a variety of existing tools are being used more and more frequently under entirely different circumstances than originally intended.

Even though these automated data analysis tools are growing more sophisticated with each generation, they still face the problem of "understanding their analyses", as fully automated search, filtering, and analysis operations are only consistently effective for clearly defined and well-understood problems.

This means that fully automated data processing methods may represent the knowledge of their creators, but they lack the critical ability to communicate this knowledge [Kei+10]. When decisions resulting from the outputs of these methods turn out to be incorrect, it is especially important to examine the procedures behind them.

2.1.2 From Data Analysis and Information Visualization to Visual Analytics

The origin of the current field of *visual analytics* stems from the automatic analysis techniques of mathematics and statistics. Developed algorithms within this field are part of a discipline with a long tradition and solid theoretical foundation. *Data Analysis* (also known as *Data Mining* or *Knowledge Discovery in Databases (KDD)*) investigates methods to automatically extract valuable information from raw data using automatic analysis algorithms [HK12; ES00; Mai06]. It deals with storing, transforming, visualizing, and processing large amounts of data from different application areas. The established techniques in this field are particularly well suited for assessing the quality of a proposed solution, which would otherwise be limited by massive data sets [HMS01; Tan+20]. The amount of data is an obvious driving problem in this regard, as irrelevant information has to be filtered out autonomously or at least before it is inspected by the user.

Hence, performance is a predominant factor in the improvement of existing data analysis techniques. This process involves a sequence of operations (i.e., data pre-processing, data mining, and data cleaning) that modify the data in different ways to extract patterns and models that reveal the implicit information contained within it. Typically, the pre-processing steps result in a dataset formatted appropriately for the data mining algorithms. According to Keim et al. [Kei+10], data mining tasks can be divided into two categories:

> In *predictive tasks*, the data is analyzed to build a global model, which is able to predict the value of target attributes based on the observed values of the explanatory attributes. An example of this is the classification of regression operations.

In *descriptive tasks*, the objective is to summarize the data using local patterns that describe the implicit relationship and characteristics of the data itself. However, existing methods support limited user interaction and are mainly designed for homogeneous data sources. Examples of this are clustering, pattern mining, association rule discovery, and many more.

Finally, the post-processing steps transform the output of the mining into a form that can be understood by the analyst. In the end, data analysis algorithms aim to provide powerful options to automatically process data sources and provide results.

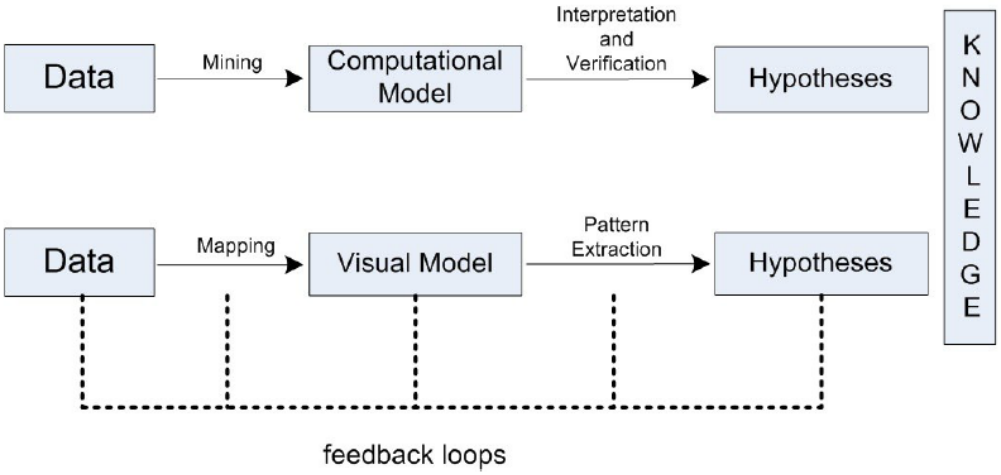

Figure 2.1: Visual representation of a comparison between traditional data mining (top) and information visualization (bottom) analysis processing according to Bertini et al. [BL10]

However, in the hands of end users, data analysis techniques often become black-box methods or result in a mismatch between algorithm and user, as the findings do not lead to a solution due to a lack of relevant expert knowledge. This is due to the fact that the early data analysis methods have been developed independently of visualization and interaction techniques from research communities dedicated to the field of information visualization [CMS99; Che06; Spe14; War13]. Figure 2.1 shows an abstract representation of these differences in analysis processing between the traditional data mining and information visualization approaches.

Outside the visualization community, *visual analytics* and *information visualization* are sometimes misinterpreted as topics with the same content. Although there is certainly some overlap, the two disciplines have substantial differences.

Traditional information visualization mostly concentrates on the process of producing views and creating valuable interaction techniques for a given class of data [Kei+08]. These visualization techniques leverage background knowledge, creativity, and intuition to address the problem at hand. However, the link to specific application areas often

causes problems when integrating developed visualization tools in new research fields. Furthermore, the developed solutions often fall short when the provided data volume exceeds what a human analyst can effectively handle, as they are intended to provide acceptable results for small datasets [KMT10].

While traditional information visualization is not necessarily concerned with analysis tasks at hand, visual analytics prioritizes visually guided data analysis in all iterations of the decision loop. Over time, more and more systems pushed to integrate the user into the analysis process. A key step in this context was the required transition from the confirmatory data analysis(using charts and other visual representations to *show results*) to the exploratory data analysis (using interactive representations to *work with the data/results*) as mentioned in the statistics research community by John W. Tukey [Tuk08]. In this regard, user interaction plays a significant role in how data can be turned into knowledge.

2.1.3 Goal and Substance of Visual Analytics Research

All changes mentioned in the previous sections lead to the development of a third approach aside from data analysis and information visualization that brings the experts' background knowledge back into the analysis process, together with the ability to interact and steer it.

An early definition for *visual analytics* by Wong and Thomas [WT04] was "the science of analytical reasoning facilitated by interactive human-machine interfaces."

Thomas and Cook [CT05] later enhanced on this initial definition as "the science of analytical reasoning facilitated by interactive visual interfaces" pronouncing the grand challenges of visual analytics in four categories:

Analytical reasoning refers to the reasoning frameworks by which users employ to gain insight or discover knowledge in order to support the decision-making process. These frameworks form the basis for applying specific transformations, visual techniques, or other operations to the data.

Visual representations and interaction covers all interactive means, methods, and techniques that enable visual representation of data.

Data Representations and Transformations address the specific ways in which data is represented, as well as the operations performed upon it (which may be noisy, incomplete, or uncertain). In this context, data representations portray the basic structure of the data within an application, which is usually not intuitive to users but serves to facilitate data transformations, computations, etc.

Production, presentations, and dissemination refers to user activity to communicate information in an appropriate context to a variety of audiences.

In order to leverage the successful deployment of these techniques, they must be moved into practice.

Keim et al. [Kei+08] agree with this sentiment and define it as "a highly interdisciplinary subject to the integral approach of decision-making, combining visualization, human factors and data analysis". This underlines the synergies of different research areas more by saying that "visual analytics combine automated analysis with interactive visualizations for an effective understanding, reasoning, and decision making on the basis of very large and complex datasets" [Kei+10]. Due to its multidisciplinary nature and wide variety of application areas, it is still difficult to find the "one right definition" for visual analytics.

However, from all of these definitions, one can clearly see the characteristics of visual analytics. The idea is to explore complex datasets through dynamic visualizations and a series of interactions by selecting, filtering, manipulating, and drilling down into the details of the data [EWS18]. Through such interactions, users explore and analyze the data in an iterative process through multiple rounds of questions and answers [Sac+14; HA08]

The driving vision is to turn the information overload (see Subsection 2.1.1) into an opportunity. Similar to how information visualization has altered our perception of databases, the aim of visual analytics is to make the processing data and information transparent for analytical discussions [Kei+10]. The goal is, therefore, the creation of tools and techniques to enable people to:

1. Synthesize information and derive insight from massive, dynamic, ambiguous, and often conflicting data
2. Detect the expected and discover the unexpected.
3. Provide timely, defensible, and understandable assessments.
4. Communicate assessment effectively for action.

In other words, the field of visual analytics seeks to provide people with better and more effective ways to understand and analyze large datasets while also enabling them to act upon their findings immediately. These "effective methods for understanding and analyzing large amounts of data in domain-related contexts" are often implemented as individual *visual analytics tools* that perform complex data analyses and prepare the results for the user at runtime.

2.1.4 Application of Visual Analytics Tools in various Domains

Over time, visual analytics spread to influence multiple disciplines for analytic reasoning and optimization such as data management, data analysis, geospatial and temporal data processing, statistics, human-computer interaction, cognition science, scientific and information visualization [And+07]. Each of these contributing research areas focuses on various theoretical and practical aspects of user assistance to solve real-world problems using information technology for efficient and comprehensible solutions. Accordingly, VATs integrate various practices of information visualization and scientific visualization together with data management and data analysis techniques while paying attention to

human perception and cognitive science. Therefore, visual analytics research can not be considered as a completely independent field due to the relationship between application context and implementation limitations, which is crucial for the development of visual analysis tools. For example Geo-Vista Studio [Gah+02; TG02] and GeoViz [HR11] address geoscience application domains, while Caleydo [Lex+10; Str+09], Cytoscape Web [Lop+10; Sha+03], or the UCSC genome browser [Fuj+11] address biomedical application domains [Lie+11]. The specific usage of these systems makes them hardly usable for any other domain, even though their visualization of analysis functionality might be helpful there.

While the goals of individual research communities may differ to some extent, there is still considerable scientific value to be gained from working closely together to initiate interdisciplinary problem-solving for joint challenges. Reflecting on the extensive range of visual analytics tools developed by Jarke van Wijk over the years, he once stated in his keynote of the final meeting of the DFG priority program for Scalable Visual Analytics

> "A team of dedicated scientists and students could develop sophisticated specific Visual Analytics applications, but still a challenging open research question remains – how to generalize?"

In fact, the generalization of domain-specific solutions into a generic one can provide a strong foundation to jump-start the development of a custom solution, which is much more efficient than creating one from the ground up.

Implementing the developing generalizable concepts effectively requires an appropriate infrastructure in the form of software and available data sets, as well as the development of a reliable evaluation methodology. Unlike other domains where first solutions exist, VA has hardly taken such approaches into account. This might be largely owed to two facts:

1. The large diversity of possibly needed tools and data cannot be foreseen for every case since the analysis process might lead to unexpected insights, which require verification with other tools offering further functionality.

2. Self-explanatory frames of reference, which provide an expressive basis for integrating other components on the view level, hardly exist in VA scenarios.

Nonetheless, there is related work on coordinating implemented visual analytics tools that address this issue from different perspectives owing to these interdisciplinary aspects.

2.2 Related Work for Visual Analytics Tool Coordination

With a well-established understanding of the definition, goals, and historical development of visual analytics research from section Section 2.1, this section focuses on the resulting properties of visual analytics tools. This includes a detailed overview of current visual

analysis systems as well as individual mechanics for coordinating different VATs. Among these are the structure and utilization of individual tools as well as their organized coupling and composite deployment in analysis processes.

2.2.1 Types of Coordination for Individual Visual Analytics Tools

In most application domains, from climatology to biomedicine, the current practice in data analysis simply invokes independent VATs one after the other [Sch+19b]. Thereby, (intermediary) results and data from one VAT are exported to the next and manually imported for further processing. The individual VATs are often arranged side by side on the same screen, while the data and intermediary results are exchanged via the file system or the clipboard. Already, this simple practice can become surprisingly involved, due to different encoding and different standards with different levels of strictness and verbosity on the data level and view levels alike. Thomas and Cook [CT05] therefore recommended the creation of "'methods to synthesize information of different types and from different sources into a unified data representation"'

Monolithic Frameworks The common answer to these challenges are frameworks, such as Obvious [Fek+11; Fek13] or VisKo [Del+11]. These frameworks provide an interoperability layer on the code level, which offers the necessary functionality to programmers and developers for coupling their VATs. Through this, individual visual analytics techniques and functionalities are combined into cohesive platforms, enabling the pursuit of a wide range of visual analyses by utilizing the standard views and computations they incorporate.

Loose Coupling A less traveled road is to follow the current practice of chaining inputs and outputs of different VATs together on the data level while manually juxtaposing their visual output on the view level. Complementary to monolithic frameworks, this approach targets the visual analysis user and assumes no access to the VATs' code base.

Lightweight Coordination Between monolithic frameworks and loose coupling, there are sometimes methods proposed as "lightweight" in terms that they coordinate where independent VATs through an underlying structure to form loose multi-tool ensembles. These ensembles offer a compromise, achieving some characteristics of integrated platforms with minimal effort, built on top of independent visual analytics tools. The features may vary from lightweight brushing and linking to the blending of different VATs into a single virtual application. Prior research on tool coordination has resulted in a variety of frameworks such as *Obvious* [Fek+11] or *VisMashup* [San+09], as well as in application-specific interoperability standards like *BioJS* [Góm+13] for the life sciences or *SAMP* [TBT15] for astronomy. All techniques in this category provide an interoperability layer on the code level, which offers the necessary functionality to software developers for coupling their VAT with other tools using the same framework to keep code changes to a

minimum and instead rely on tool synchronization on *data level* and on tool integration on *view level*.

2.2.2 From Individual Visual Analytics Tools to Coordinated Analysis Processes

Regardless of the type of coordination, the different VATs must be connected with each other in order to achieve a satisfying analysis. The previous Section 2.1 has shown that given the versatility of VATs, it is hard to adequately address all the intricacies of such connections simultaneously. Hence, the coordination of VATs (see Subsection 2.2.1) requires a differentiation between the integration of views, which is directly experienced by the user, and the integration of data, which is not.

While data is still the driving force behind knowledge acquisition, it is necessary to bring the coordination mechanisms closer to the workflow of the application domain. In order to achieve this, data-flow oriented visualization models such as the data state reference model (DSRM) [CR98] or the visualization pipeline [HM90] are still utilized as foundational elements for visualization software, including AVS [C U+89] and VTK [SLM04].

In order to provide meaningful results, this thesis follows in the spirit of Thomas and Cook [CT05] as well as Keim [Kei+10] to analyze the current state of the art based on different attributes of VATs in three fundamental columns for flow-oriented coordination (i.e, flow-oriented data management, flow-oriented analytics, and flow-oriented visualization) that are exposed to varying extend through the individual tool implementations.

2.2.3 Flow-oriented Data Management

The orchestration of data with varying quantity and quality is a key component for visual analytics, as this is typically the entry point for the data analysis [Kei+08]. However, data sources themselves are characterized by a high degree of heterogeneity, raising many challenges, and a number of methodologies, architectures, and systems have been developed to support them [Kei+10]. The subject of data management is thereby a well understood research field on its own that provides countless techniques to maintain data consistency, prevent duplication, and manage data transactions in a formal way.

Over the years, the multitude of relevant visual analytics systems brought up various methods for acquiring and managing data within a variety of proprietary or open-source solutions [VR17; Beh+19]. This dates far back to 1997, when Visage [KRL97] was presented to handle the synchronization of visualization tools through data exchange. This exchange can conceptually be understood as a three-step process of data gathering (retrieving or receiving the data from the different tools), data mapping (relating those data items that concern the same subject), and data distribution (pushing or announcing updated data to the different tools). Most of the subsequent approaches for visual analytics data management follow this principle. Nevertheless, the data exchange method can vary significantly depending on the implementation.

Centralized Data Exchange

Most approaches for data level coordination rely on a centralized mechanism. In terms of visual analytics tools, this means that there is a central entity that controls the data flow and distributes information to the connected VATs to provide uniform and transparent user access to the information stored in multiple data sources.

Database Management The traditional way of maintaining and exchanging data in software systems is to establish a shared database. The most well-known type of them are relational database management systems (RDBMS), that provide controlled and managed access based on the relational data model. This model represents collected knowledge in a database as a collection of tables that can be stored in separate files [EN16]. Tables can thereby be linked together via attribute values without explicit navigational links in the data itself, creating a clear separation between data structure and content.

Over the years, novel expressions for information sources such as streaming data [WT04; DGR05], sensor networks [Mel+06] or automatic extraction of information from extensive document collections (e.g., text, HTML, etc.) shifted the focus more towards the integration of such heterogeneous sources [Ull97; Hal01; Cal02; Dar+16]. The disciple of *data integration* deals with such structural heterogeneities to integrate data residing at different sources among multiple applications and organizations [She99; Cal+09]. There are a variety of possible architectures for data integration, like data warehouses [CD97; Inm05], federated databases [SL90] or information fusion [Bla+12], but generally speaking, most systems are positioned along the spectrum between data warehousing and virtual integration.

The goal of data integration systems is to provide a corresponding database as a central point that is assembled according to a predefined database schema. A *database schema* resembles thereby the structure of the underlying database model described in a formal language. The focus is on querying disparate data sources, which becomes a complex problem on its own if large data sources are used together. According to Doan et al. [DHI12], a typical data integration scenario involves data sources that were developed independently of each other. As a consequence, the data sources run on different systems that will have different schemata and references to objects, even when they model the same domains. Some sources may be completely structured (e.g., relational databases), while others may be unstructured or semi-structured (e.g., XML, plain text). Furthermore, the sources can change their data formats and access patterns at any time without having to notify any central administrative entity. In order to still communicate with these disparate data sources, formal languages with corresponding interchange formats such as XML, JSON, YAML, Rebol, or RDF [LSW98; Bra+08; MMM+04] have emerged and been used in a wide variety of application areas. This approach demands significantly less effort than writing and debugging the numerous routines necessary to directly convert each source schema to every target schema.

In terms of software development, a central database acts as a model in the model-view-controller pattern. Systems like *Snap-together* [NS00] and *EdiFlow* [Ben+11] both use an underlying relational database as central storage for the information exchange.

In the Snap-together, North et al. [NS00] rely on a simple API that utilizes the structure of the underlying relational database model. Visualization authors can load relations into custom visualization environments and coordinate them based on their relational joins. Thus, data-savvy users were capable of constructing coordinated visualization environments on their own using the model and interface to obtain 30-80 % performance speedup for many browsing tasks. However, the evaluation also showed that the load of multiple coordinated applications disoriented the user. Being an early adopter regarding VAT coordination, this approach implies technical knowledge about data management and information visualization to match both sides properly.

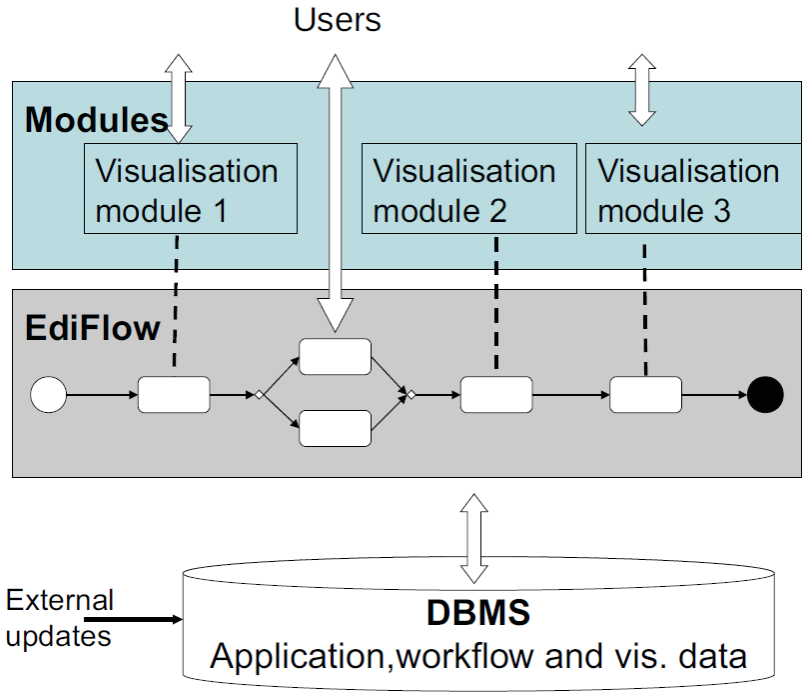

Figure 2.2: Visual representation of the Ediflow's [Ben+11] functionality as a modular service interface between the relational DBMS and the user.

In the case of EdiFlow, having a central mechanism for data handling is further utilized to add extra functionality, such as provenance information, to the process. For this cause, Benzaken et al. [Ben+11] provide a workflow platform for visual analytics to capture changes in data sources and launch a repair mechanism. Ediflow establishes itself thereby between the user and the relation DBMS as shown in Figure 2.2 The system is implemented in Java and supports the usage of SQL in a simple structured process model backed by a persistent relational database to store both process information and process instance data. The persistent management system offers a high scalability due to larger data volumes and the possibility for several users to interact with the database.

Overall, database management systems have robust means of handling security and data consistency following the so-called ACID properties (Atomic, Consistent, Isolated,

and Durable transactions). This permits both seamless concurrent data access and the option to recover data in a collection of databases physically distributed across a computer network (e.g., distributed RDBMS). As Keim et al. [Kei+10] state, while the distribution details are hidden from users, this allows access to the data via a common interface using the widely accepted SQL query language. Nevertheless, databases process transactions within seconds, and data mining algorithms available today are designed to run to completion, which can take minutes to hours or even days for complex problems.

This makes it very problematic when dealing with highly interactive content such as visual analysis results. To overcome this issue, visual analytics practitioners have begun to develop ad-hoc systems, such as in-memory databases and user-controllable algorithms [Kei+10]. However, for this domain experts cannot use off-the-shelf data storage or data mining components, and hence have to implement these systems with their often limited expertise.

Furthermore, the DBMS interface is limited to the underlying information of the database. In order to enhance the data set, either complex data integration approaches or continuous support and maintenance have to be employed. This usually requires technical experts since domain experts often lack the technical background knowledge to deploy or maintain DBMS on their own.

Service-oriented Architectures Aside from database management systems, there are also other centralized architectures that mediate not only data but also the functionality itself as services to the user interface. Service-oriented architectures (SAO) use existing tools as services to be published, discovered, and consumed by applications or other services with the goal of realizing loosely coupled, standard-based, and platform-independent distributed computing [Pv07; She+14]. Each Service is thereby used as a fundamental element to support rapid, low-cost development of distributed applications in heterogeneous environments [BSD14; Pv07; Yu+08]. This abstraction allows users to access the application's independent services without knowledge of their internal structure. This allows for seamless data exchange and the integration of specialized capabilities into unified workflows. Hence, it is often used for web applications, which are understood as web services controlled by an underlying application programming interface (API).

One key challenge for SOA and web services technology alike is the process of aggregating multiple existing functionalities into a single service, also known as *service composition* [She+02; BDS05]. This is a crucial topic, as the composition of such modular functionalities provides SOAs with the true capacity to perform more complex functions for defined user workflows. For web services, there is already a set of technologies (XML, SOAP, and WSDL) that enable the description, discovery, and invocation of a service as an independent entity. Yet, this is still missing the rich behavioral detail that describes the role the web service plays as part of a larger, more complex collaboration [She+14]. In order to achieve this, there are two common ways to describe the sequence of activities that make up a user workflow: service orchestration and service choreography [Pel03; BSD14].

Figure 2.3 shows the differences between both approaches. Service orchestration,

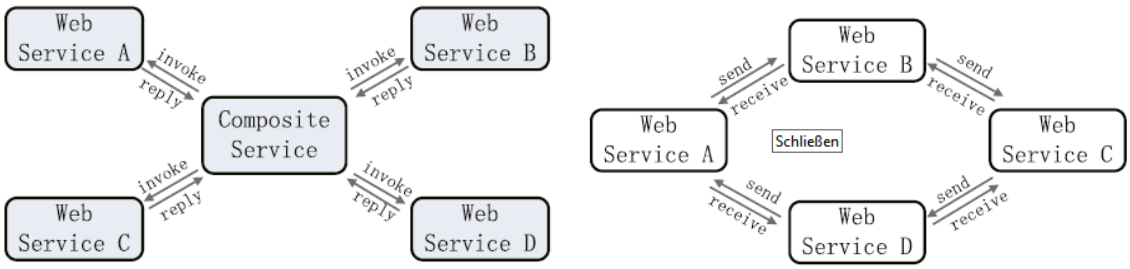

Figure 2.3: Visual representation of the structure for service orchestration (left) and service choreography (right) according to Sheng et al. [She+14]

on the one hand, refers to a singular executable process that manages the interactions among various services by outlining a flow from the perspective and under the control of a single endpoint. Thus, it can be seen as a framework that bridges an automated process with the individual services that perform the steps within that process.

Service choreography, on the other hand, represents a global description of the observable behavior of each of each service participating in the interaction. This is defined by the public exchange of messages, interaction rules, and agreements among two or more business process endpoints. It can be associated with interactions occurring between multiple web services rather than a specific process that a single party executes.

Aside from web service compositions, SOAs have similar approaches in different variations to structure a combined use of functionalities [BHL01; MBE03; MM04b; SGS04; Baï+19]. However, not all of them consider the composition of VATs, which limits their significance for this thesis, as the composition of different visualization methods is particularly important for the visual analysis process.

An example that does consider VATs is the *Metadata Mapper* by Rogowitz et al. [RM11] that builds on the component-based iPlant Cyberinfrastructure through a public API to share metadata between applications and a mapper component. This allows analysts to decide how metadata from one component should be represented in another by brushing content in one visualization so that all linked visualizations will reflect the corresponding colors. The brushing uses rules for representing magnitude information for interval and ratio data, selecting a set of colors to represent categories and ensuring legibility of text and glyphs. This introduces a design for communicating metadata from one independent component to another so that information about user selections can be mapped interactively between visualization and analysis components.

Another example is the *Plant@Hand* framework by Aehnelt et al. [Aeh+13; ASU13], which embeds a wide variety of transient and persistent data sources that each come with their associated tools for monitoring, analysis, and planning. The central interface of Plant@Hand is a three-dimensional digital twin designed for developments in Industry 4.0 to enable their shared use of machine data. To achieve this, Plant@Hand relies on data integration through enterprise service bus (ESB) technologies according to the contextualization of the applications, which provides knowledge about the flow of the data. The use of ESB technologies allows a flexible configuration and transfer of data

Figure 2.4: Screenshots of different examples showcasing visual data analysis use cases for factory visualization (left) and ship condition analysis (right) within *Plant@Hand* [Sch+20]

between heterogeneous interfaces, data models, and applications. Thus, Plant@Hand is noteworthy for its ability to combine a wide range of data sources and tools in a smart factory scenario, including enterprise resource planning (ERP), manufacturing execution systems (MES), product data management (PDM), production data acquisition (PDA), and enterprise content management systems (ECMS). The wide variety of available standards allows Plant@Hand to model multiple application scenarios from different use cases within the industrial monitoring domain, seen in Figure 2.4

However, access to manufacturing process data is more often restricted by policies to prohibit information leaks. Hence, for accessing data from different sources a complex and time-consuming installation and configuration of the overall analytics stack has to be performed to employ a matching data integration middleware. Another example architecture that closes this gap is the *OpenTOSCA* framework by Zimmermann et al. [Zim+17] that supports different data formats and protocols for the transformation and conversion capabilities in order to make the accessed machine and factory data processable by the analytics service. It leverages the OASIS standard Topology and Orchestration Specification for Cloud Applications (TOSCA) standard to model analytics algorithms as smart services, enabling automatic provisioning and management of the analytics and data integration stacks through Apache Tomcat, Flink, and some additional Python-based analysis libraries running on a Ubuntu virtual machine.

Independent of the specific architectures, service-oriented architectures still have to deal with the interoperability between different components of multiple services. This is usually done through a communication protocol over the network. Hence, there are some deviations in the service level communication depending on the protocol or the network structure.

As Daraio et al. [Dar+16] mentioned that since data and processes are packed into services, it is insufficient to explicitly make the meaning of data and processes. Instead, services become other artifacts that require documentation and maintenance, which

complicates the governance issue. Similarly, data warehousing techniques, which promote a distinction between data management at the operational level and data at the decision-making level, do not resolve this challenge. In fact, they further complicate the system by duplicating data across different layers and introducing synchronization processes between those layers [Len11].

Decentralized Data Exchange

In case no centralized data exchange mechanism is provided, tools use custom connectors to pass data between them. This could be for example achieved through tool-specific APIs by receiving and sending requests. If no connectable interface is provided by a tool, the exchange can still be achieved through minimally invasive methods for data retrieval.

Lightweight Broker If data from more than a few tools must be coordinated, a lightweight broker or data pool is often established to ease the pairwise integration and to provide a minimalistic data mapping of identical data items [HMS03; Di +09; Ali+11]. This is similar to the approaches mentioned in Section 2.2.3 just that instead of relying on a central database that holds all data sources, a decentralized source management is applied with a lightweight broker that channels the stored information to the tools that request it.

As traditional data integration approaches are schema-based, they are not adequate to deal with dynamic situations [HMS03]. They suffer from a high front development effort to analyze and encapsulate data sources, determine a global schema, and require precise schema mapping. Therefore, mashup systems are used to extend the agility of the dynamic data integration of multiple sources.

Mashups A mashup system aggregates functionality, presentation and/or contents from existing web sources to create a new application. The content is usually generated by either using web feeds or an API [Ali+11]. At its core, a mashup system is yet another web application that aggregates multiple services to achieve a new purpose [XCZ15]. Its structure is similar to the three components of the Model-View-Controller (MVC) pattern [Wol94; Max+07]. Most mashup systems are commonly built by encapsulating data sources as data services with a unified format and then aggregating these data services into a composite one. This offers the required agility and expressive power for non-professional users to build a situational application from multiple data sources as it frees the users from the burden of defining a global schema and complex schema mappings [Han+13]. However, in order to instantiate such an internal data model from an external data source, the mashup tool itself must provide strategies for the mapping of said sources.

Some specific versions of mashups called *data mashups* allow users to access, process, and combine data from various sources by generating queries and executing sub-queries using XQuery [Ali+09] and SPARQL [PS+06] with data sources in XML and RDF format [LSW98; MMM+04]. According to Di Lorenzo et al. [Di +09], the internal data model of a data mashup system is a single global schema that represents either an object-

or graph-based unified view. In an *object-based model*, the internal data is structured according to objects. Thereby, an object follows the standard for object orientation and is defined as an instance of a class. The class contains the features of an element, including the characteristics of the element (attributes, fields, or properties) and its behaviors (methods). In a *graph-based model*, the graph refers to the model based on XML and extensions of it such as RDF or RSS. This design decision is driven by the reality that most of the data accessible on the web is formatted in these ways. Therefore, most data mashups use a graph-based model. Nevertheless, the results of either of both approaches can be published as web services, APIs or native applications depending on the specific implementation [HMS03].

One popular example is the Mashroom system [WYH09], that relies on nested tables to encapsulate multiple data sources, which the unified data model adapts for internal processing and external uses. The internal data model is structured to define the content and representation of data items in a relational model [Han+13]. Data is mapped based on the instance data instead of the common approach of directly mapping onto the schema. The component access model of Mashroom defines the data formats and access protocols supporting JSON, XML and HTML as data formats that can be handled by HTTP, REST and SOAP [Han+13; PZL08]. Mashroom+ [LWH14] extends on functionality to improve matching correctness and reduce user effort through a semi-automatic mapping approach. Since precise semantic mapping is hard to establish with automatic matches due to the uncertainty of potentially imprecise data schemata [LWH14], Mashroom+ includes an interactive matching algorithm. With this, the automatic matching results can be synthesized from multiple matchers as well as from expert users feedback.

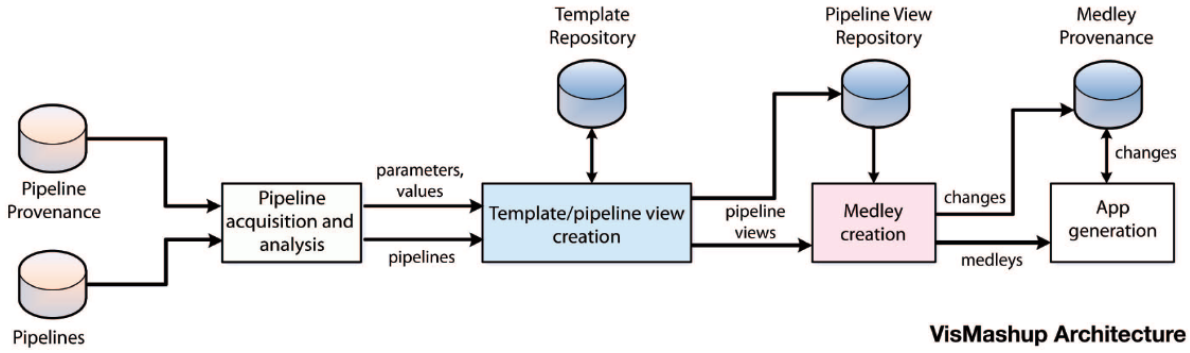

Figure 2.5: Visual representation of the data flow in the VisMashup pipeline architecture by Santos et al. [San+09]

Another example is the VisMashup system by Santos et al. [San+09], which uses a set of components to support the tasks of an application designer. The architecture of this system is essentially described by four components to handle the data flow as shown in Figure 2.5. Thereby a pipeline is referred to as task, that a designer needs to carry out in order to develop an application from mining and exploring a set of visualization specifications.

- The *pipeline acquisition and analysis component* (PAAC) allows the designer to

query and mine pipeline collections and their provenance.

- The *template/pipeline view creation component* (PVCC) enables the designer to manipulate the pipelines and create simplified views, which expose only task-relevant pipeline components.
- The *medley creation component* (MCC) combines multiple pipeline views from the PVCC into a medley for the creation of more complex and exploratory tasks.
- The *app generation component* (AGC) uses the medley specification from the MCC to automatically generate the application and the associated user interface.

The underlying structure is based on the data flow model from Lee and Parks [LP95]. This data flow model is widely used in visualization systems, including ConMan [Hae88], AVS [C U+89], SCIRun [PJ95; PWJ97], and VTK-based systems such as Paraview [AGL05; Cal+08], VisIt [Chi+11] and VisTrails [Mor+13]. Essentially, the data flow model employs two operations within the pipeline: execution (which runs a pipeline) and substitution (which modifies the components of the pipeline). This is suitable enough to be used during the application design and execution.

Other specific versions of mashups called *ubicomp mashup* go far beyond the limit of APIs and web services and use four types of components for data retrieval: electronic hardware, mechanisms, and physical phenomena, local code in the form of off-the-shelf software and remote code in the form of web infrastructure services [HDK08]. With these mashups, sensors and actuators can connect with existing or repurposed mechanisms, as well as other physical phenomena like embedded programmable microcontrollers. This allows designers to create their own programs or utilize off-the-shelf software on their personal computers. These local applications may provide hooks for programmatic automation via APIs or built-in scripting languages."

Direkt Information Extraction Due to the diversity of standards and formats for data encoding and data description (i.e., metadata), many existing VATs limit themselves to one particular application domain – even though, their underlying approaches would be generally applicable. If access to such VATs is not accompanied, alternative means of accessing isolated information may have to be considered. Data retrieval from otherwise closed tools relies on minimally invasive methods, like screen poking and screen scraping, to yield the desired data [Fer+11; HDK08].

Computational Steering is the interactive control over a computational process during execution [Mvv99; Sch+16]. This control is usually facilitated by a visualization of the progress or an intermediary outcome of the computation with which the user can interact. The information about the running process is extracted and displayed at so-called checkpoints, breakpoints or sync points during the computation. These points along the running process can be either predefined or interactively set by the user.

Search for Identities is commonly utilized in visualization and UI research to map data from different applications [Fou+14; Wal+10]. However, the result of this approach must often be replaced by more complex mechanisms in real-world applications.

Dovetail joints align and integrate two components through a shared interface for a unified user experience. Similar to woodworking, it stresses precise fitting, unlike loosely combined services. This approach is deliberate and well-planned, creating an enhanced user experience. Examples are documented extension and integration points in the system architecture-APIs in software, breakout headers and connectors in electronics, and mounting holes in hardware [HDK08].

Hot Glue is used when combinations of components appear incompatible, lack mutual awareness, or do not inherently support each other. This approach (much like applying actual hot glue) comes with limited adhesive power, offering only superficial, surface-level integration. It emphasizes a quick, temporary fusion that holds elements together without deeply integrating their functionalities and symbolizes a practical yet transitory linkage of services that might not naturally harmonize.

Screen Poking refers to the generation of synthetic mouse and keyboard events by web programmers. This method is employed to remotely control software and extract information from diverse sources. It serves as a technique predominantly utilized in web programming to interact with software interfaces and obtain data. Just as poking implies a targeted interaction, screen poking involves specific actions that simulate user inputs, allowing programmers to remotely navigate and retrieve information from different software interfaces.

Screen Scraping is the process of extracting information out of HTML documents to harvest information from online databases. This is done by processing either the DOM-tree or parsing rendered user interfaces [Fer+11]. This involves defining an extractor or wrapper to select the relevant information out of the DOM-tree [Kus00] or a vision-based approach that attempts to provide a more general solution to the problem by assuming that similar content types have similar visual features [LMM06; Cai+03]. Examples of toolkits for scraping data from different sources are the combination of firegoose [Bar+07] and the Gaggle Tool Creator [TBB10] or the combination of SideCache [DBR11] and SideKick [DYR10] which cater specifically to the biomedical domain.

2.2.4 Flow-oriented Analysis

While the data management component of visual analytics tools is mainly involved in the storage, transport, and traceability of data, the analytic component takes said data as input and processes it to produce output in the form of diverging interpretations of information. This can be done either automatically or by the user depending on their expertise regarding the provided functionalities.

As mentioned in Subsection 2.1.3, visual analytics solutions are always developed for a specific use case. This use case is based on an existing problem of the user, which results from his workflow. Thus, the workflow is also the driving force for the analytical exploration of the underlying data and, therefore, the baseline for the overarching analysis process.

From User Workflow to task-oriented Analytical Process

Workflows are generally described as a sequence of steps that lead to results. However, this is a very vague definition. Therefore, workflows are more often understood as repeatable patterns of business activities used to execute processes. During the office automation era of the 1970s and 1980s, analytical process management systems began to gain traction, leading to the emergence of various terms in the 1990s, such as business process modeling (BPM) and business process engineering (BPE) [Alo+97; van+03; Zur04]. Subsequent digitization and growing interest in optimization established workflow processing in both new scientific and commercial areas. Today, analytical process management systems are divided into two main categories: business analytical process systems and scientific analytical process systems.

Business workflow systems typically resemble flowcharts, state transition diagrams, or UML activity diagrams, highlighting an execution model centered around control flow patterns and events, while data flow is often considered a secondary concern [Lud+06]. Examples of it are commercial systems like Inforsense's DiscoveryNet [Ćur+02] or the Pipeline-Pilot [War12].

Scientific workflow system usually features execution models that place a greater emphasis on data flow. Examples of it are academic systems including Karma [SPG08], Kepler [Lud+06], Taverna [Oin+04; Bel+08] and Triana [Maj+04]. The Kepler system [Lud+06], for instance, aims to support a wide variety of workflows, from low-level plumbing tasks to high-level analytical knowledge discovery and conceptual design.

Regardless of whether it is a business workflow system or a scientific workflow system, there are different approaches to workflow optimization based on the specific application scenario. In the case of visual analytics, the workflow is usually represented by the user's tasks to achieve a goal by utilizing the provided interactive visualization of data. This goal can be represented by one of the three distinct goals for visualization as described by Keim et al. [Kei+06].

Presentation aims to efficiently and effectively communicate the results of an analysis. The input is based on facts that are fixed a priori. The challenge of this task is the choice of the appropriate presentation technique in order to emphasize the structure behind the data.

Exploratory analysis is the process of searching and analyzing data to find implicit but potentially useful information. Interaction with the data is needed to reveal insightful information. Findings in visualizations can be used to steer model building in the automatic analysis. Supporting this task is quite difficult since the analyst has no hypothesis about the data.

Confirmatory analysis can be described as a goal-oriented examination of one or more hypotheses. These hypotheses about the data serve as a starting point in order to confirm or reject them as a result of the analytical process.

Out of all the visualization goals addressed, the exploratory analysis of data is of particular importance concerning visual analytics tasks, as the domain user is usually not interested in the type of visualization chosen but in the conclusions drawn for their workflow. Exploratory visual analysis is, therefore, a well-established iterative method in which analysts start with an overview of the data to pursue various hypotheses through multiple rounds of interaction and analysis [EWS18].

Understanding flow-oriented Behavior based on Pipeline Operators In its most typical form, the visualization process adheres to a pipeline of operators. These operators can be adjusted before or after being applied to the data, but not while they are used to generate the visualization. This behavior resembles the classical term of visualization pipeline as described in Schulz et al. [Sch+16]:

> A *visualization pipeline* models the visualization process by encapsulating algorithmic steps in operators that are usually depicted as rectangular nodes. The passing of data between these algorithmic steps is modeled through transitions that connect the nodes in the form of directed edges. Each operator is responsible for transforming data from one data state into another. These states are commonly termed incoming (raw) data, derived data, geometry data, and image data. The transformation processes between them are usually called filtering (incoming data), derived data, mapping (derived data), geometry data), and rendering (geometry data) image data). The DSRM extends this base model to permit operators to transform between data states and operate within a data state.

This monolithic approach to visualization works well for traditional scenarios where the input and output constraints remain consistent. However, such stable scenarios can no longer be assumed, as these constraints are often more fluid and difficult to identify throughout the visualization process. No matter what the goal of a particular domain application might be, the visual analytics workflow can be regarded as a task-oriented analysis.

Categorization of Analysis Functionality

There are several environments available for task-oriented analysis reaching from simple applications or components over integrated solutions or web-services to fully-fledged

toolkits or program libraries. This section follows a similar subdivision according to Keim et al. [Kei+10], who included various application scenarios such as:

- Statistical analysis (e.g., SPSS [Webe], SAS [Webg], R [Tea])
- Scientific computation (e.g., Matlab [Webc], Scilab)
- Textual analysis (e.g., GATE, UIMA, SPSS/Text, SAS Text Miner)
- Image analysis (e.g., Khoros [Arg+00], IRIS Explorer [Fou95])
- Video analysis (e.g., OpenCV)
- Machine learning toolkits (e.g., WEKA, scikit-learn, NumPy)

Through this analytical functionalities can thereby be classified into different categories depending on the degree of user inclusion and visual information representation through the visual analytics tool.

Statistical and Mathematical Tools Statistical analysis has a long history of visualizing the results as time series, bar charts, plots, and histograms. Examples of tools providing statistical and mathematical visualization are R [Tea], Matlab [Webc], Mathematica [Webj] and SAS [Rod23; Webg] for statistical computing and graphics.

Specific Algorithmic Tools Algorithmic tools have been developed by the research communities for a specific task or problem. Examples are Graphviz [Ell+02; Webb], open source graph visualization software, or Pajek [BM02; Webd], which places a greater emphasis on analyzing social and complex network data by utilizing network and graph visualization.

Visual Analytics Libraries One example aimed initially at providing data visualization and visual design expertise is BirdEye [Weba], a community project to advance the design and development of a comprehensive open-source information visualization and visual analytics library.

Visual Data Mining Tools Visual data mining tools create visualizations to reveal hidden patterns in datasets. The need for new methods in data analysis has launched the field. Several solutions in this regard are often centered around business intelligence areas such as marketing, risk analysis, sales analysis, and customer relationship management.

An example of this is the Konstanz Information Miner (KNIME) [Ber+08b; Ber+08a]. It is a modular data exploration platform that enables the user to visually assemble data flows, selectively execute some or all analysis steps, and later investigate the results through interactive views on data and models. KNIME uses the Model-View-Controller (MVC) pattern [Wol94] with a centralized database, that provides access to the nodes, with processes input and output of data or model in terms of basic operations such as

filtering or merging to simple statistical functions (e.g. computations of mean, standard deviation or linear regression). However, as KNIME is a data scientist tool, the inputs for each workflow step are unclear and the visualizations are not as neat and polished as some other open source softwares. (e.g. RStudio [Rac12])

Another popular visual data mining tool is the open-source Waikato Environment for Knowledge Analysis (Weka) [MR06; Webi]. It includes a suite of machine learning algorithms for data mining tasks, enabling users to create pipelines for data pre-processing, classification, regression, clustering, association rules, and visualization.

A further well-known environment for machine learning and data mining tasks is RapidMiner [RBP15; Webf]. It enables users to design data flows that encompass input and output, data pre-processing, and visualization. Additionally, it incorporates learning schemes and attribute evaluators from the Weka learning environment.

Web Tools and Packages An ever-growing array of tools is accessible online. Still, user interaction becomes more complex and challenging to model and optimize when accessed remotely. These tools enable users to generate visualizations using their own data. One example of an online analysis tool is ManyEyes [Vie+07], an IBM application for social data analysis. Another example is VisFlow [YS17], a web-based system that enables users to connect various data processing and visualization components using a drag-and-drop interface to configure data analysis workflows. VisFlow aids the user in this regard by providing a library of data transformation and analysis operators for manipulating and analyzing datasets.

On-Line Analytical Processing Over the years, on-line analytical processing applications have gained increasing acceptance for displaying business visualizations, as they encompass not only purely representative data visualizations but also techniques for advanced interaction and visual querying [SM09].

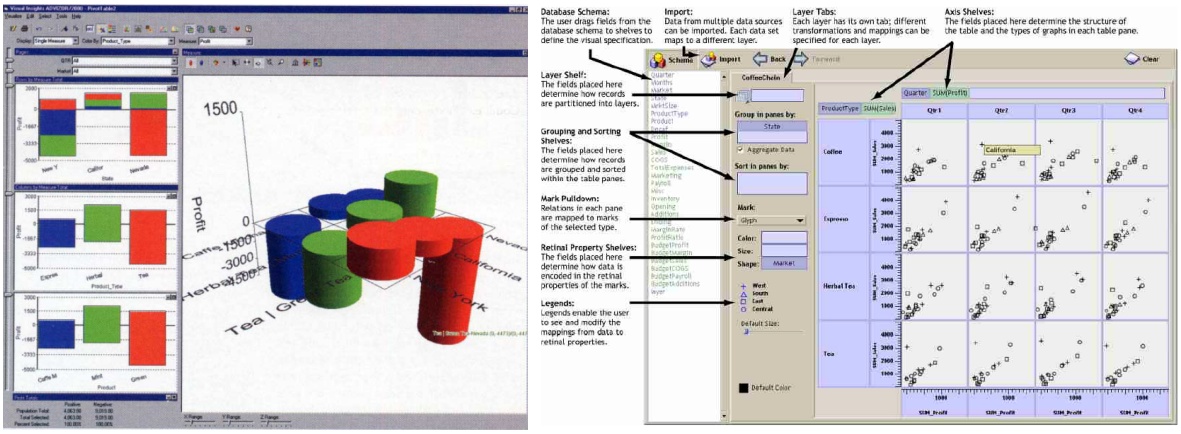

Figure 2.6: Screenshots from the on-line analytical processing systems Advizor by Eick [Eic00] (left) and Polaris by Stolte et al. [STH02] (right).

The term On-Line Analytical Processing (OLAP) [CG98] refers to end-user applications

for interactive exploration of large multidimensional datasets (usually residing in the organization data warehouse). These applications utilize a multidimensional data model to explore data from various perspectives using data cubes (or hypercubes). Data cubes improve upon the traditional spreadsheet format by automatically aggregating and sorting data, with the results stored in a secondary table, known as a pivot table (called a pivot table), so that they may hold millions of entries characterized by multiple dimensions of hierarchies. This allows the end user to explore them through traditional or rather specialized visualization techniques such as time series plots, scatterplots, maps, treemaps, cartograms, matrices, decomposition trees and fractal maps.

Advizor and Polaris (shown in Figure 2.6) have been among the first attempts in this direction. *Advizor* [Eic00] arranges multidimensional data from multiple tables onto a series of pages, each one containing several linked charts. The interactive data visualization and in-memory data management facilitates ad-hoc exploration of the data. The at Stanford University developed *Polaris* [STH02] system inherits the basic idea of the classical pivot table interface using embedded graphical marks rather than textual numbers in the table cells. The types of supported graphics are arranged into a taxonomy comprising rectangles, circles, glyphs, text, Gantt bars, lines, polygons, and image layouts. Stolte et al. successfully commercialized the pioneering Polaris [STH02] and VizQL [Han06] research through the well-known Tableau visual analysis software [CSH03].

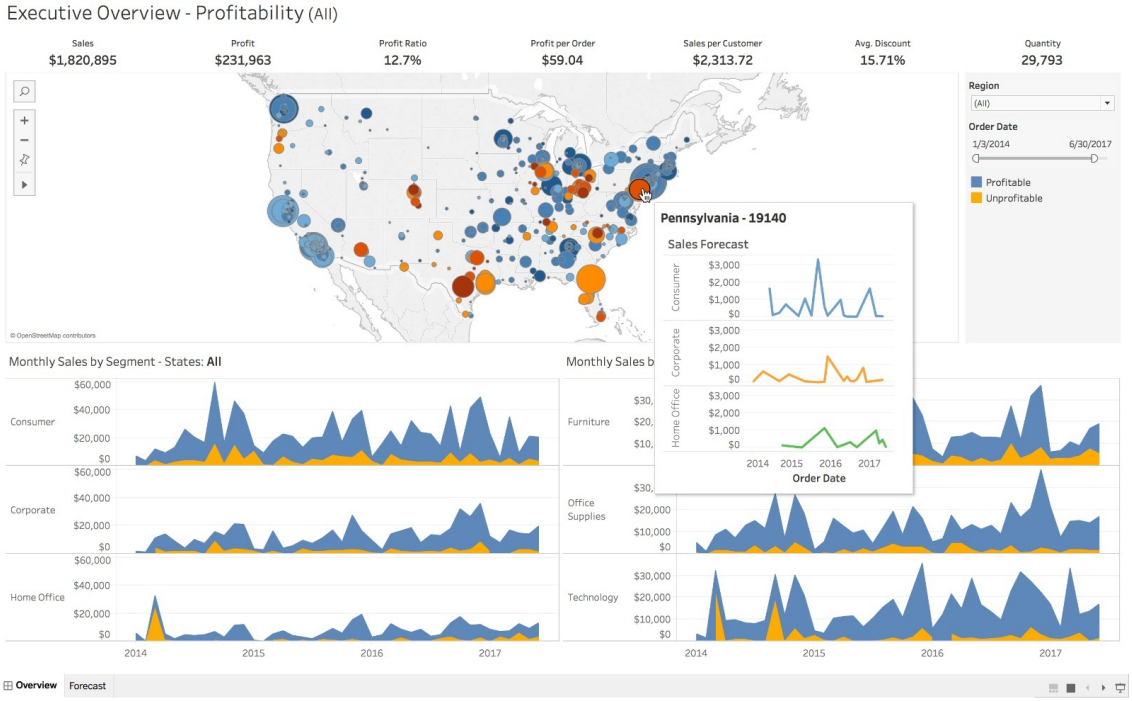

Figure 2.7: Screenshot of the Tableau [CSH03] user interface showing an example of the executive overview of profitability visualized with multiple charts.

Nowadays, Tableau (see Figure 2.7) is among the leading platforms for the visualization

of modern business intelligence and is widely recognized in a wide range of application areas [Lot19; Bat+20].

The biggest challenges for OLAPs are:

1. *Maintaining a high degree of interactivity:* This can only be achieved by pre-computing and storing aggregated values for various levels within the hierarchies. This reduces the size of the data and, therefore, increases the performance at the cost of lost precision.

2. *System Usability:* The user can only explore a limited number of dimensions at once, as the hypercube must be projected onto two- or three-dimensional spaces. Gaining insights into high-dimensional data may involve lengthy and sometimes frustrating exploration.

Situational Applications Situational applications refer to the applications that can be constructed on the fly by business users and/or departmental IT staff to deal with transient and ad-hoc requirements. Such user-centric applications not only process data from enterprise back-end databases, but also deal with Web data from HTML pages, web services or open APIs, which are not covered by traditional enterprise data integration architectures.

2.2.5 Flow-oriented visualization

With the data management component handling transport and traceability of data and the analytic component handling the production of results, there is still one key component of a visual analytics tool left that resides in the interactive visualization of results. Not only is this part responsible to show what the user is working on, but it also enables the interaction with the information on screen. Therefore, the visualization component within VATs is much more than the "simple" display of graphics to communicate data values. It further supports the monitoring of operations, the guidance of users, or the steering of processes in disciplines related to data management and data mining.

Representation of Analytical Results

In order to present the information of an analysis on screen, visual analytics tools follow the basic approach of visualization tools and present the data either as scientific visualizations or information visualizations.

Scientific visualization is primarily concerned with the accurate representation of real-world phenomena such as fluid flows or molecular structures from the world of engineering, biology, or medicine [Kei+10]. The goal here is to represent data as physical entities such as surfaces, volumes and flows in terms of time and space using a wide variety of visualization techniques such as scalar, vector, or tensor fields [SM00; EK04; Eng+06; HJ11]. Common problems in this area are particularly related to the efficiency

of the representation for interactive exploration and the automatic derivative handling of relevant visualization parameters.

Information visualization , on the other hand, is concerned with the representation of abstract data for which no explicit spatial or temporal reference is required [KW02; CMS99; Spe14; Aus08]. These typically include business data, demographic data, or network graphs, which often span hundreds of dimensions and have no natural mapping to the representation space [Kei+08]. Since standard visualization techniques such as plots, line, and bar graphs are rather ineffective for this purpose, the focus in this research area is predominantly on developing novel visualization techniques such as parallel coordinates [ID91], treemaps [Shn92], glyph- [FSL05] and pixel-based [KKA95] techniques to reduce the clutter of the displayed information [ED07]. Problems in this regard exist especially with respect to the versatility of the application domains and the typically vast amounts of data, of which most have to be pre-processed by automatic analysis techniques such as clustering or classification.

While visual analytics is applicable in both areas, most of the use cases are related to information visualization. This is due to the more common practice of using abstract representations in analysis processes to represent data. Thus, knowledge can be communicated faithfully without enforcing a one-to-one representation of the collected data. When it comes to the final display of information, there are countless approaches to visualization that might be relevant depending on the respective end-user task. Apart from pure representation, however, interaction with the displayed content is an elementary component of both scientific visualizations and information visualizations.

Reaction to Changes in the View Space

When it comes to interactive visualization (especially when using multiple application windows), the portrayal of changes remains crucial throughout. Following the background definition of Schulz et al. [Sch+16] display strategies can be divided into two categories, assuming that out-of-core visualization and computational steering are part of the progressive character of user interaction in analytical workflows.

Parallel visualizations on the one hand, distribute the visualization process across multiple processing units and execute it concurrently [Ahr+01; Ahr+07; Vo+11]. This parallelism can be implemented by partitioning and distributing the data (i.e., data parallelism) and/or by subdividing and distributing the visualization process itself (i.e., pipeline parallelism) [Mor13; Vo+11]. Additional caching mechanisms [Ben+09] and scheduling mechanisms [BP08] can be used to enhance the parallel execution by allowing the re-use of stored results from previous computations and avoiding redundant calculations, respectively.

Progressive visualizations on the other hand, aim to continuously provide visual updates even during lengthy data transmissions [RS09; SPG14], data loading [GKW14],

and rendering procedures [Fre+14] for which one would otherwise have to wait. Its goal is to generate and display partial yet meaningful results from ongoing processes. This architectural paradigm utilizes data chunking by segmenting and re-arranging the data to be transmitted or by sampling the data to be rendered in progressively larger chunks. Incremental visualizations, applied during analytical tasks, are, however, more complex to set up, as an iterative refining visualization process gradually adds or updates visualization details to complete the visual representation of a dataset. Therefore, analysis tasks must either be orchestrated directly with the user or predicted and steered.

Combination of Interfaces in Conjunction with the User's Analysis Task

Aside from the type of visualization and their reaction to changes over time, the most pivotal point for visual analysis is the association with the user's analysis task. This requires user interfaces to follow the described flow-oriented process in conjunction with the display of information during various steps of the analysis, which can be achieved in several variations.

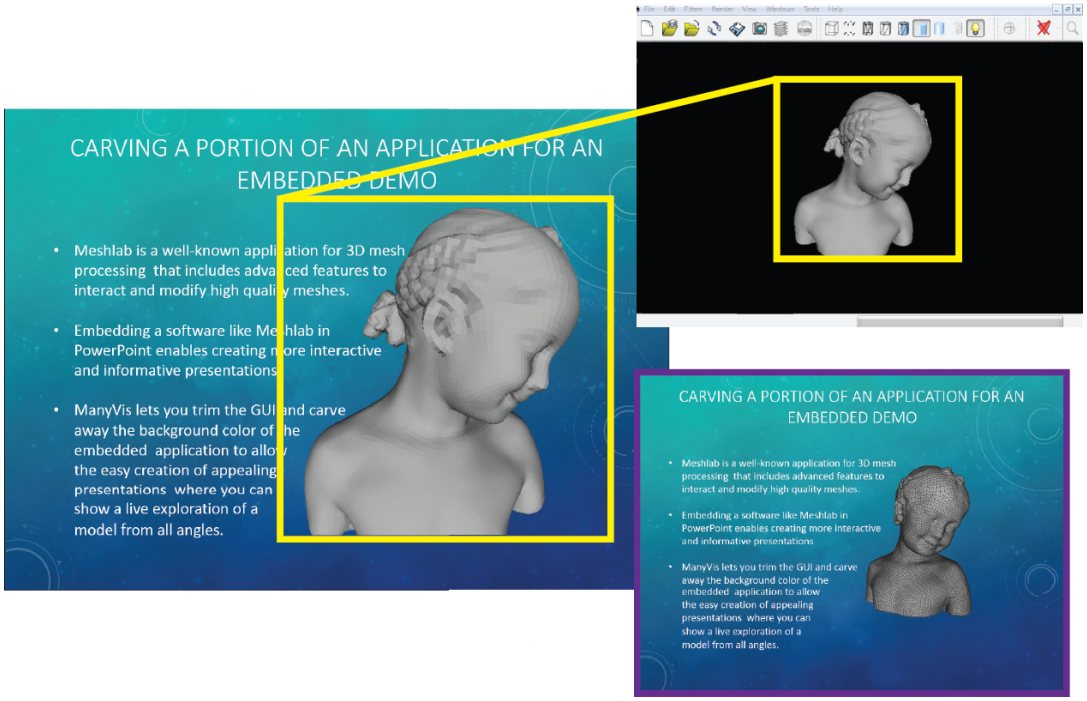

Figure 2.8: Screenshot of the ManyVis application by Rungta et al. [Run+13] showing a fully integrated MeshLab [Cig] demo by cropping the unnecessary GUI while maintaining full interactivity and applying alpha transparency to the embedded application (purple inset).

Exchanging UI Components One method to achieve a task-oriented representation of the process is through the exchange and integration of user interface (UI) components and visualizations from various systems, as suggested by approaches like ManyVis by Rungta et al. [Run+13] (shown in Figure 2.8). Since the coordination of applications at the data and view levels is independent, they can be combined as needed to provide the necessary flexibility for a specific analysis.

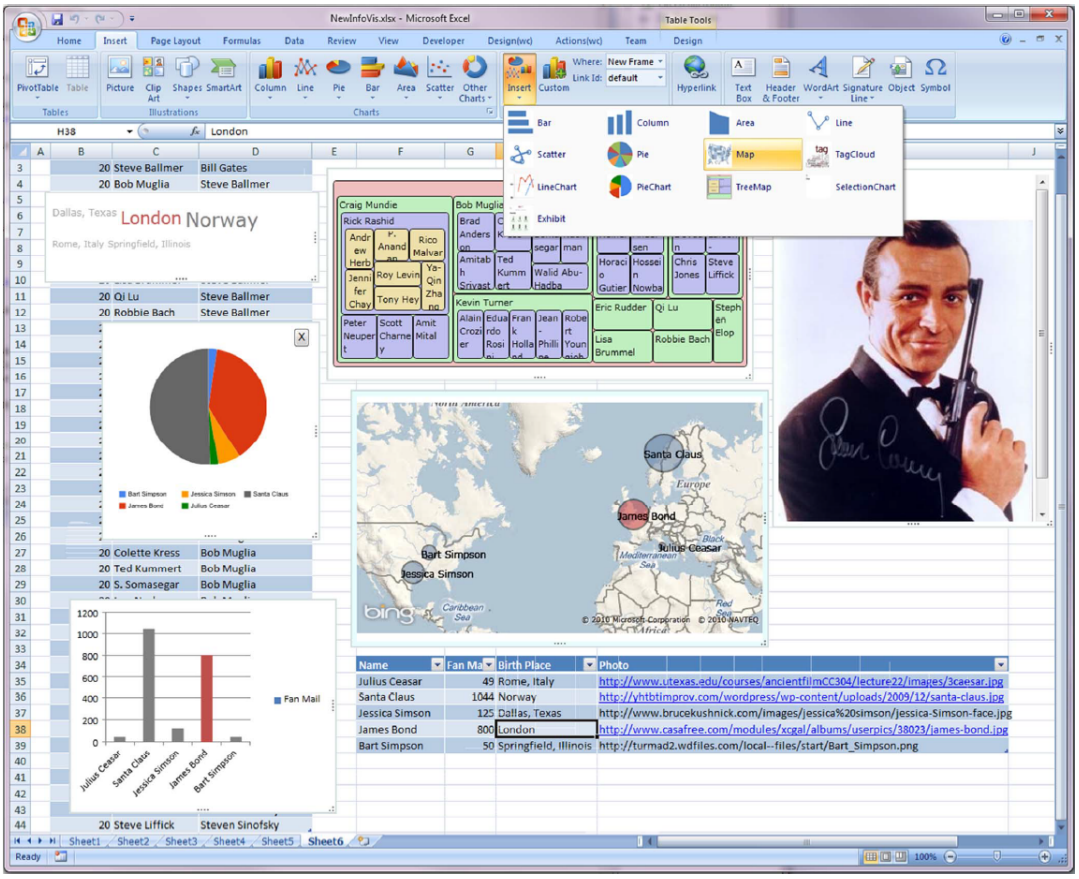

Figure 2.9: Screenshot of an Excel spreadsheet that displays the usage of WebCharts by Fisher et al. [Fis+10] for the visualization of framework tag clouds, maps, tree maps or images on top of the underlying window.

One example of such coordination on both levels is Webcharts by Fisher et al. [Fis+10] showing various windows of framework tag clouds, maps, or tree maps in Figure 2.9 together with focus and context visualization on top of a spreadsheet. To make the interoperability of the tools available to the user, standard visualization frameworks usually display such an interlinked setup in multiple coordinated views [Dör+08; Rob07] or fused visualizations [Nor+03]. In some cases, the toolchain itself is visually encoded in a (directed) graph layout that shows their connections [Nor+02; TIC09; WN13; Gür+16]. Yet, in some cases, arranging UIs side-by-side is not sufficient, and a common interface may be glued together from individual interface parts.

Reusing UI Parts When arranging UIs side-by-side is not sufficient, a common interface may be glued together from individual interface parts [Ali+08]. *WinCuts* [TMC04] and *Façades* [Stu+06] enable users to interactively compose their own UI by cutting out pieces of existing applications and re-arranging them into one tailored UI. Approaches like AppMap [RGF11] can help identify frequently used UI elements. Whereas, Prefab [DLF11; DFW12] and PAX [Chu+15] automatically detect UI elements to alter them or to add interactive enhancements. For web-based ensembles, existing UIs are combined in mashups [PND10; Par+15], ranging from search result visualization [Dör+08; Spo07] to technical information about structure [Nat98] and use of webpages [Sch67].

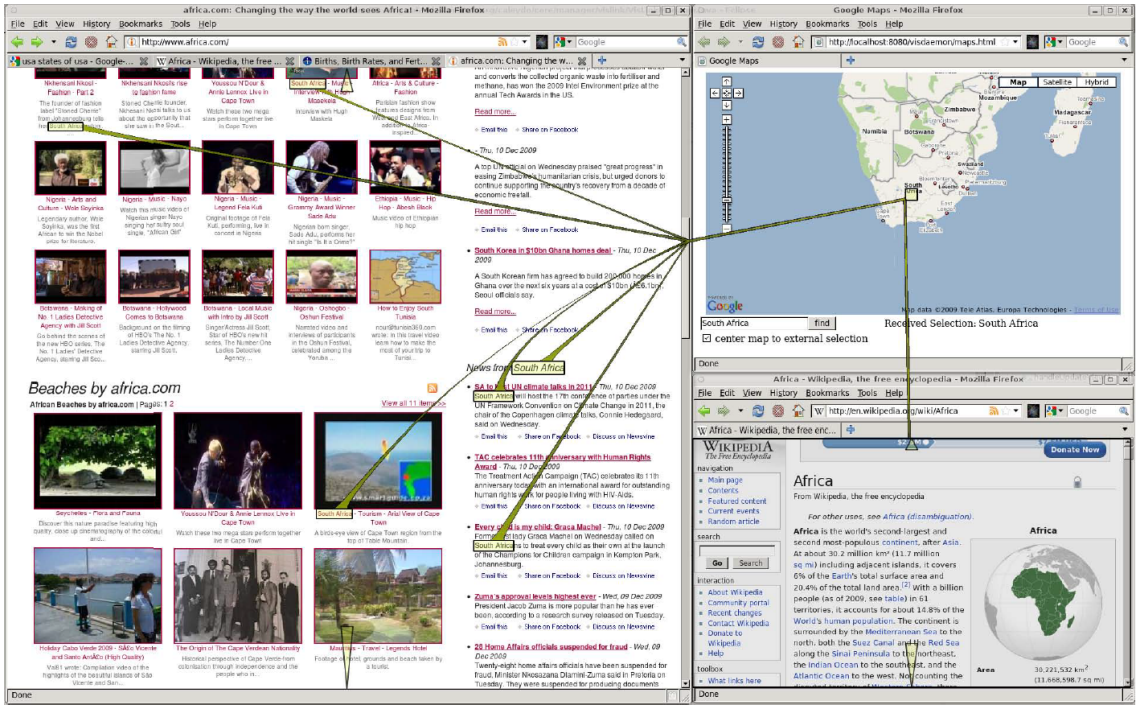

Figure 2.10: Screenshot of the visual linking approach from Waldner et al. [Wal+10], where the related information is highlighted in different views across multiple applications

In the approach from Waldner et al. [Wal+10], visual links are used to connect related pieces of information across application windows in a minimally-invasive interface for cross-application communication. As shown in Figure 2.10, the user's attention is thereby visually guided towards the relevant information. The flexibility to highlight features and see their projections in other data views ("brushing") broadens the analysis process since it lends itself to finding patterns in the entire collected data, not only for some selected data points [RM11].

InterTwine from Fourney et al. [Fou+14] aims to establish an "inter-application information scent" through widgets and search snippets by linking relevant information from the web browser in the native desktop applications as shown in Figure 2.11. This simplifies the user's finding and re-finding information within and across multiple

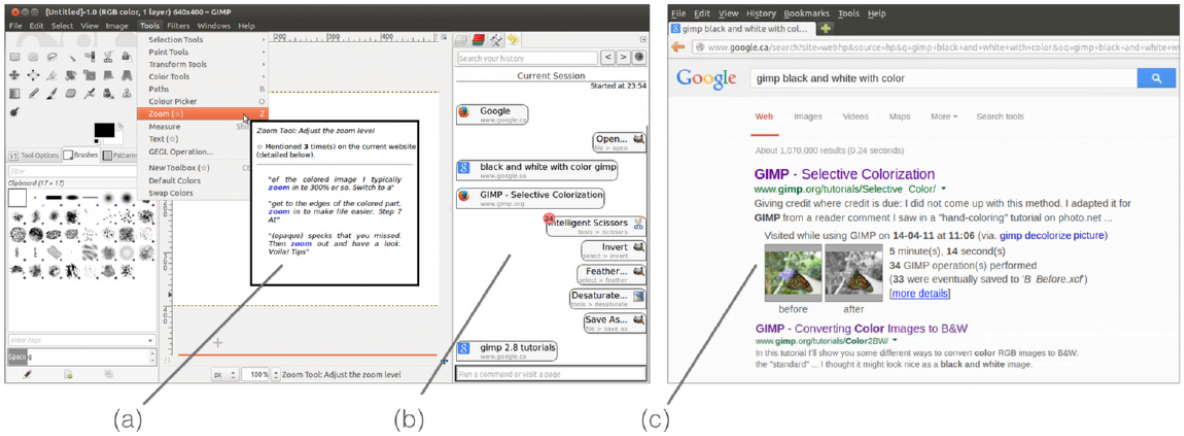

Figure 2.11: Screenshot of InterTwine from Fourney et al. [Fou+14] showing GIMP and a web browser side by side as they are interconnected through (a) tooltips to describe how the GIMP is mentioned on the current web page, (b) an interactive event history to facilitate re-finding past actions and (c) web search snippets with details of how work documents evolved when that page was last accessed

applications. InterTwine is composed of three conceptual entities:

- A *shared inter-application history* produced by recording actions in the web browser and desktop application
- A *filter mechanisms* for the identification of relevant information in the inter-application history
- An *interface mechanisms* with context-aware, inter-application information scent synthesized from the shared inter-application history

The application explores these concepts in the context of GIMP by adding an inter-application history transcript, embellishing its menu items with beacons and adding history snippets to tool tips as well as augmenting Google search result pages in Firefox with history digests [Fou+14].

Relying on Uniform UIs Another possibility for achieving a task-oriented representation of the process with more seamless visual interoperability is to replace individual UIs with a uniform interface. In many cases, however, users are accustomed to particular UIs, so replacing them would more likely decrease usability. Therefore, some systems re-use interface parts to bridge tool functionality instead of creating yet another competing toolkit.

One example approach in this direction is the meta-toolkit Obvious [Fek+11], which relies on the idea that all the classes to implement data tables in software libraries are very similar (if not identical) except for variations in method and class names. Therefore,

it abstracts class definitions that serve as a pivot to share data for all encapsulated Java libraries by mapping 1) a native data table to an Obvious data table and 2) the tables in the other direction. For this, Obvious uses wrappers that are defined for four visualization toolkits, two machine-learning toolkits, and the standard Java API for database access. In this way, it offers tangible and immediate advantages to visual analytics application developers by enhancing the reusability and interoperability of code and software components while postponing the decision on which toolkit(s) to use until a later stage of the visual analytics application. Furthermore, it enables toolkits and library developers to incorporate their tools into the extensive ecosystem of obvious-compatible systems. Obvious is based on the InfoVis Toolkit (IVTK) [Fek04] that implements an in-memory database manager, where data is organized in columns, contrary to most persistent relational databases, to improve the memory footprint, allowing the addition of new attributes that are needed to manage the interaction (e.g., selection or filtering) and to hold attributes computed on demand. The main challenge is supporting interactive performance for rendering and dynamic queries with a small memory footprint. The Obvious data model is centered on a proxy tuple design pattern [HA06] to enable high extensibility and good usability as it encompasses graphs in an object-oriented manner [Fek13]. Therefore, Obvious improves reusability and interoperability of software components.

Another approach by Streit et al. [Str+12] used the workflow of an analysis task as a trajectory through data to create a unified representation of the analytical process shown in Figure 2.12. The resulting Stack'n'flip application contains detailed information on suitable visualization and computation methods and provides orientation by inferring possible paths within the modeled analysis setup. The application itself is based on three different underlying models:

- The domain-independent **Model of the Setup** enables the interactive visual analysis by describing the available data sets and their interrelations, the available visual, and computational methods and packages, as well as their analytical operations. This model is authored once and needs to be adapted or extended only when new data sets or tools become available.

- The **Model of the Domain** captures what can be done with a given setup in the context of a specific domain, describing the numerous domain-specific tasks and relating them to the data sets and analytical operations of a given setup model. A certain setup model can be associated with different domain models, as different application domains may use the same setup to carry out the analysis. Examples are in the field of life sciences, where a geneticist and a biochemist may use the same data sources and interfaces, but perform completely different tasks.

- The **Model of the Analysis Session** lists what has to be done to pursue a given analysis goal, describing the analysis workflow as a sequence of domain-specific tasks from a given domain model.

The authoring process used is based on a data model that captures all datasets available in an analysis setup, including local data sets (e.g., an electronic patient file),

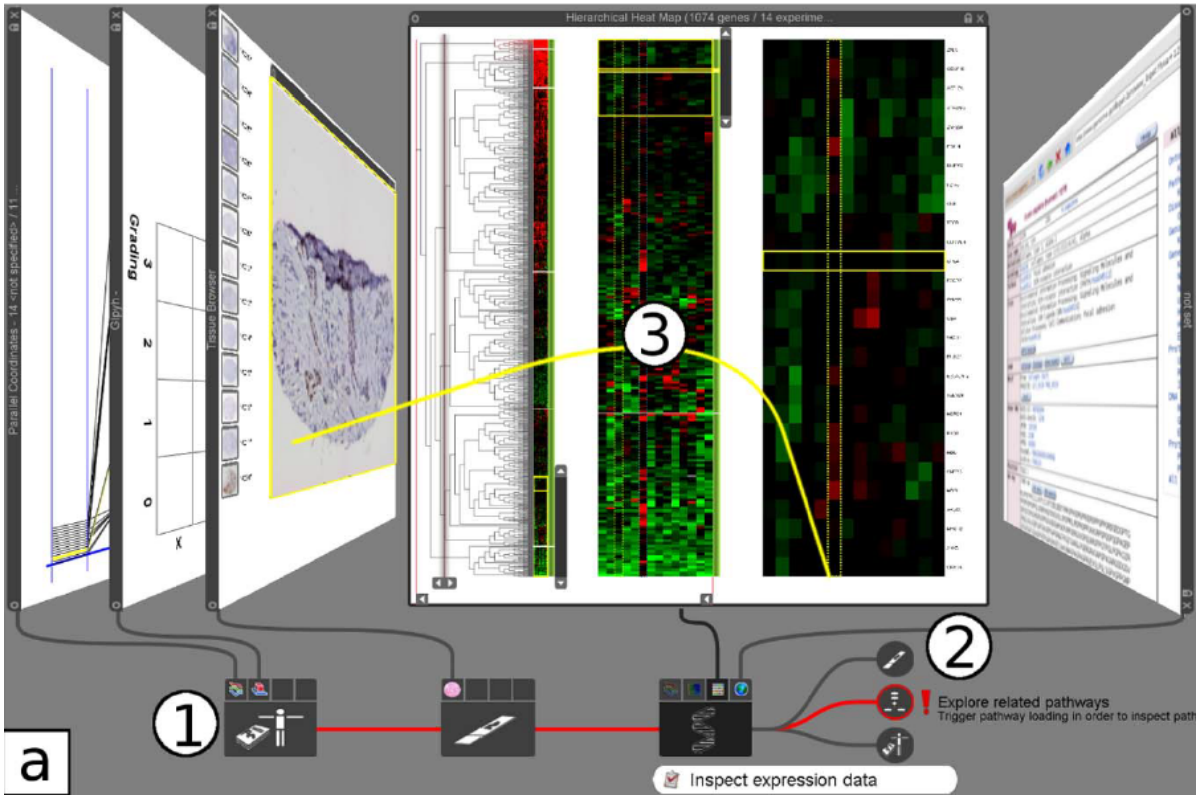

Figure 2.12: Screenshot of the Stack'n'flip application from Streit et al. [Str+12] with (1) showing the succession of the analysis path taken on different data sets, (2) showing possible future steps or branches and (3) emphasizing relations between individual views through visual links

data sets available from online databases (e.g., pharmaceutical lists or digital anatomical atlases), or streamed data (e.g., a patient's vital signs coming from intensive care). These datasets can contain different types of data, such as imaging data from body scans, gene expression data from micro-array analyses, or text data from electronic documents. Therefore, common keys or identifiers are used to relate the different datasets to each other (e.g., the patient's name or social security number for the identification of different patient records). The visual interface enables an automatic determination of suitable analytical techniques and subsequently uses this information to provide navigational cues on a much more specific lower level along a given analysis path. This proves especially useful in interactive systems that exhibit many possible continuations at any given point. This complex model offers essential benefits for highly repetitive analysis sequences as they can be re-used for routine tasks in the set use case. However, the overhead of such an elaborate modeling phase is not justified for straightforward setups with a manageable complexity [Str+12].

Coordination for the Layout of Views

Many VATs base their design choices on Shneiderman's famous mantra for visual data exploration "Overview first, filter and zoom, Details on demand" [Shn03] that describes how data should be presented on screen. However, due to the typically large volumes of data in visual analytics, generating an overview visualization can be challenging without overlooking significant patterns. This renders zoom and filter techniques nearly ineffective, as users lack guidance on how to further explore the data. Hence, Keim et al. [Kei+06] extended this mantra to shift the focus more toward the underlying analytical approaches as well as the management of the available information to "Analyse first, Show the Important, Zoom, filter and analyze further, Details on demand". This indicates that it is not sufficient to solely retrieve the data and display it using a visual metaphor, but rather that it is necessary to analyze the data according to the value of its interest, showing the most critical aspects of the data while providing interaction models that allow the user to retrieve details on demand [Kei+10]. At the center of this process is always the user, who interactively manipulates both the data and the representation based on their pre-existing knowledge. Hence, it is necessary to control the visual representation of the user workflow.

Dashboards A common way of displaying several visual representations of different content in a flexibly configurable overview is dashboards. As El Meseery et al. [EWS18] defined it, dashboards are often portrayed as "a visual display of the most important information, consolidated and arranged on a single screen so the information can be monitored at a glance" [Few06; Zha+12; EB11]. In terms of visual analytics, a dashboard is therefore considered as a "faceted analytical display, which holds multiple charts, together with the interaction techniques to support efficient analysis" [Few06]. Well-known commercial systems like Tableau [MHS07] or SAP Lumira [Dut16] use this presentation technique to display as much business information as possible given the spatially limited display area of an average office screen. Although multiple dashboards can be created, these systems maintain a consistent filter state across all of them. VizDeck introduced a drag-and-drop approach for constructing dashboards based on recommendations [Key+12]. Voyager [Won+16] later employed a breadth-first strategy by displaying several recommended charts in the initial view, allowing users to gain an understanding of their data. This was continued in Voyager 2 [Won+17] by adding facilities for drill-down, filtering, and related actions. While dashboards provide an overview of the data and permit analysts to explore it, they offer only a single-state interface, meaning that any modifications to the filtering or selected dimension will alter the entire system view.

Multiple Coordinated Views The important factor for this visualization comparison discussion is that these coordinated views can be algorithmically linked in such a way that actions and highlights in one view can be reflected in other views. According to Collins and Carpendale [CC07], these Multiple Coordinated Views (MCVs) [Dör+08; Rob07] allow for the re-use of the spatial visual variable, granting each relationship

type its own spatial representation. The temporarily activated visual connections offer a significant advantage over manually locating related data items. Yet, the relationships themselves are not visually represented. In some cases, the sequence of workflow tasks itself is visually encoded in a graph layout that shows their connections [NS00; TIC09; WN13; Gür+16].

Multiple Tab Interfaces Contrary to the approach of splitting the whole user interface into multiple parts, some studies focused on web applications that featuring a common navigation bar through multi-tab interfaces [HW10; Cha+21]. Although utilizing a web browser to open multiple versions of an application allows for various states and versions of the data, a single web application with multiple tabs does not offer the same advantages as multiple coordinated views shown at the same time. Yet, employing multi-tab interfaces is still a viable solution for managing limited screen space. One study demonstrated that, compared to traditional multi-tab browsing, employing a tabbed interface with clear support for user goals decreased disorientation and facilitated information gathering [LC10].

DynSpace The DynSpace from El Meseery et al. [EWS18] is a multi-workspace VAT with a multi-tab interface allowing users to create workspaces that are independent of one another while obtaining the interaction in the same way. It supports both open-ended exploration and focused data analysis through multiple coordinated views for each workspace, view recommendations and manual view specification by providing thee main panels: information about the dataset and their attributes, workspaces and their views or charts and bookmarks showing previously saved charts. A single workspace is thereby defined as a fully editable dashboard or container that holds multiple coordinated views at the same time with the associated state (i.e., data filtering and visual encoding) and one or more analytical goals (i.e., data wrangling, exploration, clustering) [EWS18]. This allows users to see multiple states of their work and thus support different analytical tasks. Large high-resolution displays combined with small mobile displays appear to be particularly suited for collaboration [Hor+18; ASS19]. However, according to Brehmer et al. [Bre+22] supporting seamless interaction experiences in such cross-device scenarios is a challenge for future work.

Display Ecologies The area of display ecologies (or smart environments) considers four important design aspects in this regard: display composition, display membership, information coordination, and information connection [Chu+15]. Multi-display environments can support users in gathering, combining, displaying, and connecting content parts from different applications [Rad+15; ENS15] in a framework for smart view management [RLS11]. Based on suitably described data [Sch+16], views, and tasks [Sch+13], these approaches have been employed in a use case from the domain of visually analyzing climate change data from different providers [Eic+15]. It bundles and instantiates much of our prior research on adaptive interfaces and tailored visualizations, such as [HDK08; Abe+14; Mar+14].

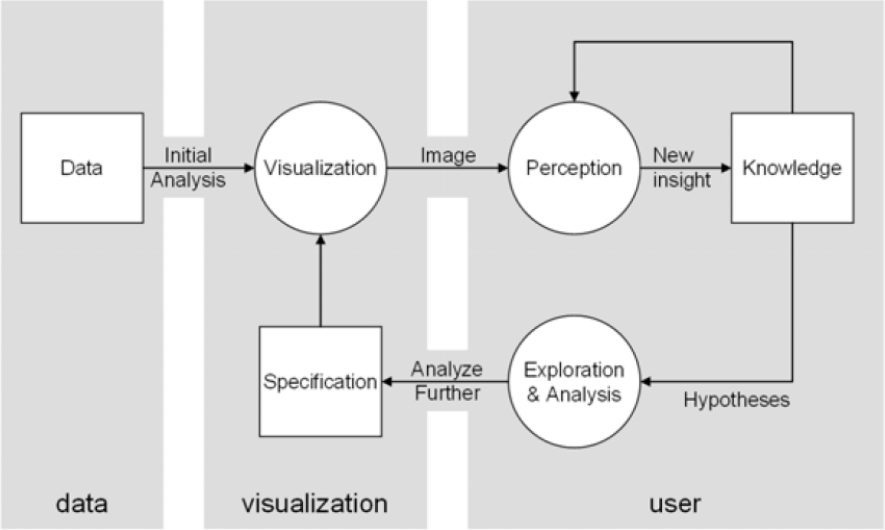

Figure 2.13: Visual Representation from Keim et al. [Kei+08] for the sense-making loop for visual analytics based on the simple model of visualization by Van Wijk [van05]

2.3 Summary for the State of the Art

Overall, visual analytics can be seen as an iterative process that follows the idea of a "sense-making loop" as it is described by Van Wijk [van05] in which the user continuously gains insights about the provided data (see Figure 2.13). This knowledge discovery is, for the most part, designed for the specific use cases in which alternating between visual and automatic methods leads to continuous refinement and verification of preliminary results [Kei+10]. The required tools to extract such knowledge from data can, thereby, range from pure data mining to complex visual exploration frameworks that each achieve their goals through different structures, functionalities or visual representations depending on the desired use case [Kei01]. The in-depth investigation of these systems showed that coordination between these existing structures, functionalities, and visual representations is desired, as solid efforts have been made to facilitate the use of created systems and algorithms.

Subsection 2.2.3 demonstrated that numerous data management approaches, including databases, direct library linking, component integration, inter-process communication, and service-oriented application architectures, align with the objectives of visual analytics. However, they are usually intended to support the connection without the explicit use of visualizations, strategic analysis, or decision-supporting processes in mind. Hence, even the fastest data management system is cumbersome if the task of the analysis deviates from the actual purpose of the underlying architecture.

Subsection 2.2.4 showed that there are many existing tools with a vast variety of statistical, algorithmical, analytical or mining functionality to gain information from the input data. However, there is a significant discrepancy in the level of services provided

due to the lack of fast, imprecise answers with progressive refinement, incremental recomputation, or steering towards interesting data regions that are of higher interest to the user. It is purely in the user's hands to understand the data output and gain knowledge from the provided information, which becomes increasingly difficult if visual interactive support is missing.

Subsection 2.2.5 showed that the visualized results of the analysis can be shown parallel or progressively through different layout strategies such as dashboards, multiple coordinated views, multiple tab interfaces, dynamic space management, or display ecologies. Furthermore, the resulting views of such layouts can be combined or associated by exchanging UI components between applications, re-using existing UI parts in a combined visual structure, or relying on an overarching uniform frame of reference. While this helps in order orchestrate the view space, it is rarely used in coordination with the specific tasks of the application domain. The problem of this thesis is, therefore, somewhat similar to the challenges of display ecologies in that it requires a coordination of UI components and data from different VATs based on the user-specific domain tasks. This leads back to the design issues identified by Chung et al. [Chu+15] in which the user must transfer different data and views to coordinate information and link particular views.

All in all, there are many valuable approaches for the connection and interplay of data, function and views spread across multiple disciplines. The focus of each visual analysis scenario seems to be bound to the interactive visual representation and exploration of data. With the goal of creating a solution that integrates the necessary and couples the possible, this thesis therefore consider these separations and coordinate among different layers to achieve an interlinked perspective for next-generation VATs.

3 Problem and Requirement Analysis for Visual Analytics Tool Coordination

In this chapter, the problem description from the introduction is examined with the background in mind to declare the challenges of this thesis for individual VATs as well as for the overall visual analysis process. These challenges are then discussed in relation to the research questions to define hypotheses. Finally, requirements are defined to explore the hypotheses that provide the framework for the following conceptual model.

3.1 Defining Challenges based on Problem Analysis and Objective

As seen in the previous Section 2.2, current approaches towards tool coordination mainly deal with specialized solutions or ad-hoc workflows comprising multiple VATs and custom scripts to support application domains such as medicine, earth and life sciences, manufacturing, or business analytics. Each of these tools contributes very specific and often domain-dependent functionality to the visual analysis that is not provided by generic off-the-shelf analysis frameworks, such as SPSS or Knime. Relying on a single superapplication to address all of a user's analytical needs is, hence, unrealistic. Consequently, there is a strong need for bridging the gaps between existing, independent VATs [Str+12].

Instead of proposing yet another new one-stop framework, such as Plant@Hand3D [ASU13] or KNIME [Ber+08b; Ber+08a], this thesis aims to validate the analytic procedures to embrace independent domain tools and build upon them. At the same time, the resulting approach is meant to provide options for including generic frameworks in the pool of available VATs, thus enabling their coordination with the special-purpose VATs that originated in the application domain. Having special-purpose and general-purpose VATs available under a unified user interface will bridge the gap between them, remove adoption hurdles, and permit fluid back-and-forth switches as needed. In this way, this thesis charts a path towards combining existing VATs into novel, tailored visual analytics solutions that are more than their individual parts. Therefore, the goal is to develop general coordination concepts and mechanisms at the data and view levels that can readily be transferred to various application domains.

3.1.1 Investigating the Space of Visual Analytics Challenges

Considering the objective of this work, challenges for coordination can be derived from the individual problems of VATs in the visual analytics process. As seen from examples in Section 2.2, the visual analysis of various data sources can generally be understood as a repetitive, layered process that undergoes data management, analytics, and visualization in an interactive manner. Creating a comprehensive interface that integrates multiple otherwise individual VATs into such a process is, therefore, a multidisciplinary task. This is especially true if the resulting solution should be independent of any domain, as various aspects of VATs need to be considered.

Keim et al. [Kei+06; Kei+08] addressed the issue of application independence and tried to find measurements for individual VATs in order to support the analytical process, resulting in the following collection of upstanding visual analytics challenges:

Scalability : Processing large datasets requires (1) means of computational overhead or optimization to ensure performance and (2) appropriate rendering techniques to display the results while maintaining an overview.

Heterogenity : Information from numerous heterogeneous sources requires the synthesis of heterogeneous types of data in order to perform an effective analysis.

Interpretability : Generating a visually correct output from raw data and drawing the right conclusions requires a high quality of the used data and analysis methods. During the analysis process, possible quality problems in the data (e.g., errors, noise, outliers, low precision, missing values, coverage errors, double counts) should be avoided at all costs.

Problem-Solving : Real-world problems are manifold and therefore in-transparent, consisting of conflicting goals, and complex in terms of large numbers of items, interrelations, and decisions involved. Detecting these problems and finding solutions requires a well guided human-computer-interaction.

Semantics : Semantic meta data extracted from heterogeneous sources may capture associations and complex relationships. Therefore, providing techniques to analyze and detect this information is crucial to visual analytics applications. This requires increased capabilities for creating and maintaining large domain ontologies as well as methods for the automatic extraction of semantic meta data, since the integration process between different ontologies to link various datasets is hardly automated.

User Acceptability : Many novel visualization techniques have been presented, yet their widespread deployment has not taken place, primarily due to the user's refusal to change their working routines. Therefore, advantages need to be communicated to the audience of future users to overcome usage barriers and eventually tap the full potential of the visual analytics approach.

Integration : Tools and techniques should not function in isolation, but rather, they should integrate seamlessly into the applications of diverse domains and facilitate interaction with other already existing systems.

Evaluation : Evaluating visual analytics approaches is considered challenging, as it necessitates specific criteria on how to judge a visual analytics solution or application. While there are established criteria in the separate fields, which visual analytics seeks to draw together, it is difficult to conceptualize how these could be cohesively aligned. Various aspects can be examined in this context, including functional testing, performance benchmarks, measurement of display effectiveness, economic success, user studies, and the evaluation of impact on decision-making. For instance, in the field of visualization, numerous techniques and criteria exist for the evaluation of results, such as assessment of the effectiveness of the result (through user evaluations). Likewise, it is relatively easy to judge the outcome of traditional data mining approaches through validation of the results with reference data. However, in terms of integrated visual analytics solutions, it remains unclear what constitutes a 'good' solution or how an application should be designed.

Overall, many of these aspects do not only relate to the VATs themselves but also to problems related to human-computer interaction (HCI) and human perception science. Looking realistically at the listed longstanding challenges, it becomes apparent that a comprehensive solution would not be feasible within the scope of one thesis. However, it highlights an important factor that permeates all challenges: the interactive nature of visual analytics applications. Since the analytics process is always user-driven, this particular aspect should be considered when defining challenges.

3.1.2 Separation of Concern for the Interactive Visual Analysis

In order to incorporate the user-driven aspects of visual analytics for the challenges in the coordination process of VATs, this thesis aligns with the inverse measure of directness for interactive visualization tools, as articulated by Tominski and Schumann [TS20]. Directness is thereby described as a crucial factor that significantly influences the smoothness and efficiency of the action cycle and the user's ability to engage deeply in an interactive dialogue with the data. To gain a comprehensive understanding of 'directness,' Tominski and Schumann state that it is valuable to explore this concept from an opposing perspective. Essentially, directness is inversely related to the degree of separation between the human's action and the systems's responses, meaning that high degrees of separation hinder the achievement of directness.

Conceptual Separation relates to the different models, such as the user's mental model, the system's implementation model, and the interface's represented model [CRC07] that are involved when interacting with VATs. While the mental model comprises the analytic problem being dealt with and the concept a user has about the system to abstract from details and focus on goal-relevant aspects, the implementation model adheres to a formal description of technical details, algorithmic conventions,

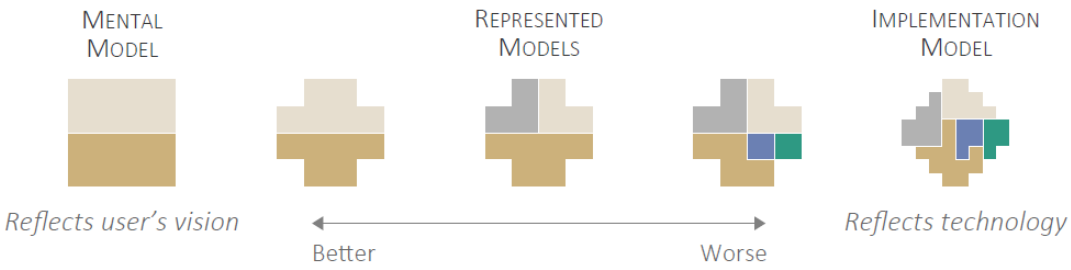

Figure 3.1: Visual representation for the conceptual separation across different models by Tominski and Schumann [TS20], which was adapted from the original concept by Cooper et al. [CRC07]

and parameterized procedures. Both models exhibit a large conceptual separation, as illustrated schematically in Figure 3.1. On top of that, the represented model captures what the user can actually see and interact with on the display. The closer the represented model is to the mental model, the more directly users can interact with the system.

Spacial Separation concerns distances to be covered as the interaction costs increases when users have to move the pointer over a larger distances in order to execute certain actions. Furthermore, it is problematic when the eyes have to switch frequently between different parts of the screen when evaluating the system's response. Considering the scattered data visualization and the corresponding graphical user interface mentioned by Tominski and Schumann [TS20] (see Figure 3.2), it becomes apparent that action and effect are spatially separated. This can make it more difficult to understand the action-effect causality. The user may need to switch their focus between the user interface and the main view multiple times in order to understand how certain parameters affect the visual representation of the data.

Temporal Separation is about the latency between the user's action and the system's visual response. When latency becomes excessive, it adversely degrades the efficiency of interactive visual exploration [LH14]. Ideally, a response should, therefore, be provided within 50–100ms [Shn94; Spe14]. Nevertheless, the computations required for processing the visualization transformation and generating visual feedback can take a considerable amount of time. An example of this would be the visualization depicted in Figure 3.2, which includes scattered data points, their corresponding Voronoi diagram, and a continuous color gradient in the background. While rendering the individual data points and computing, the voronoi diagram can be computed relatively quickly (given the small dataset of merely ten data points), the smooth coloring could cause a noticeable delay leading to the adverse effects of efficiency degradation as color values need to be computed for every pixel of the main window. For a moderately sized window with a 1,280 x 1,024 resolution, this would already include 1,310,720 pixel changes,

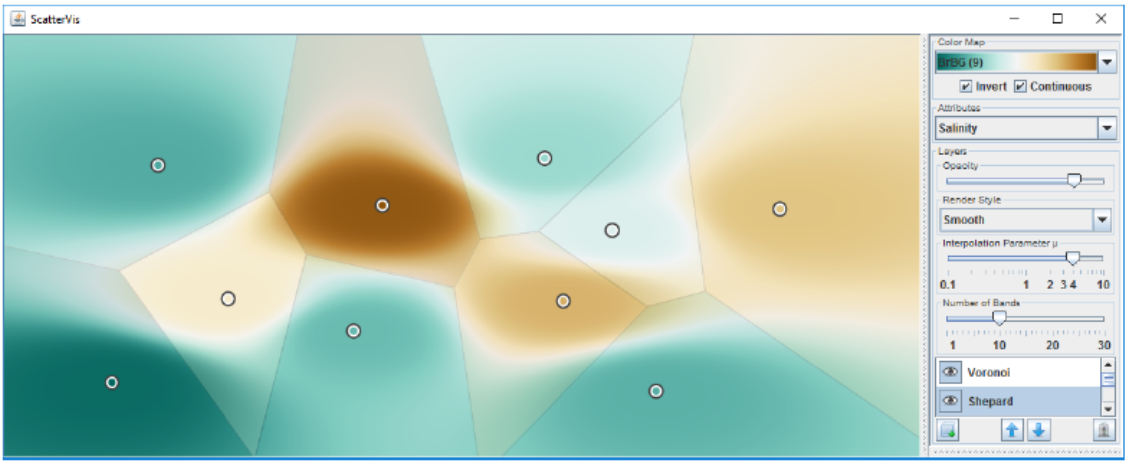

Figure 3.2: Screenshot of ScatterVis from Tominski [TS20; Webh] displaying the spatial separation between the graphical user interface (right) and the visual representation in the main view (center).

As Tominski and Schumann [TS20] stated, "all three types of separation impair directness and hinder smooth and efficient visual data analysis". Consequently, the minimization of separation should be a central focus throughout the entire process of visualization development, reaching the design as well as the implementation phases. This separation of directness provides the fundamental definition for the challenges that are important for the scope of this thesis, as both data and tools must follow the demands of unspecified analysis flows.

3.1.3 Challenges for the Coordination of Visual Analytics Tools

Following the previous discussion from Subsection 3.1.2, the challenges covered in this thesis are defined by the conceptual, spatial, and temporal separation of interactive visual analysis to harness the power of independent coupling between pre-existing sources on data and view level. This concerns both the model and the implementation of such coordination methods. Overall, the design of interaction and visual feedback should aim to minimize both conceptual and spatial separation, while the implementation should provide sufficient flexibility for algorithmic efficiency to reduce temporal separation.

Customizable Configuration of situational Processes As previously mentioned, domain experts provide the tasks for VATs, while technical experts perform the implementation. Therefore, it is reasonable to assume that domain experts do not carry as much weight on the underlying implementation as technical experts but rather on the fact that data can be traced back along the toolchain so that information is presented reliably throughout the analysis. This conceptual separation of applied usage, data management, and functionality throughout the situational analysis process poses one of the hardest hurdles to overcome when dealing with independent software solutions. The innumerable

amount of technical implementation formats for data sources further limits the need for a customizable configuration of such tasks.

Hence, adjustments regarding performance and comparability might be necessary for the efficient use of the toolchain. Such potential data transformations or conversions require a flexible interface with different options for the modification of persisting information. Depending on the data exchange capabilities of the VATs involved, this might be done automatically by the system or manually by a technical expert on site.

Scalable Data Exchange and Screen Space Management The next big challenge for VAT coordination is the fact that the amount of data items can vary quite a lot throughout a visual analysis resulting in limitations for the implemented systems as well as for the overall human perception. This is especially true for an assumed unknown amount of views that stem from individual VATs. Regardless of the physical equipment (in terms of computation speed and display size), the visual analysis is always limited by algorithmic performance and human capabilities. This spatial separation of stored data through multiple sources and displayed information through multiple views requires scalable solutions for data exchange as well as screen space management.

Approaches for the coordination of VAT should, therefore, be scalable in the sense that they can be scaled down to inform the coordination of multiple linked views in a single VAT (which is by no means a solved problem [KC16]), and that they can be scaled-up in terms of supporting multiple available data exchange mechanisms without making application-specific limitations. Due to the nature of a visual analytics process, this is not limited to the displayed space but also includes the layout over time, as the question would have to be answered for multiple workflow steps. Hence, regulations for screen space management might be required to define a specific application domain beforehand or, in general, as part of a dynamic solution. It is reasonable to assume that there are missing details about the technical structure of each VAT and their communication capabilities. To facilitate data exchange in such an environment, it is essential to offer generic options for data transport. While achieving this may not be feasible for every system, it is nevertheless desirable to provide multiple exchange mechanisms that are easy to deploy.

Temporal Overview and Process Steering Since visual analysis more often consists of more than one step, the presentation of results must adapt to the process progression. For example, a VAT in the analysis process could be removed for one step and then reinserted in the next one. However, due to the independence of the different VATs is not only an obstacle for data coordination but also for visual control over the analysis process. This temporal separation of defined tasks performed actions, and received responses is another key challenge for the overall coordination process.

To avoid the manual starting and stopping of VATs, it is, therefore, necessary to enable measures for effective steering. These should be intuitively perceived by the user and allow him to take control over the course of their analysis. An example of this could be overview mechanics as part of a directed graph structure, as in Streit [Str+12].

During the analysis, it might be helpful to reflect changes made by the user to help them adapt the pre-defined analysis process if needed and take control when necessary. It is reasonable to assume that such changes in the view layout might disorient the user during their analysis task. Hence, it seems to be necessary to guide or at least offer possibilities for orientation to support the user throughout the visual analysis process.

3.2 Building Hypothesis based on the Research Questions

This section follows up on the research questions described in the introduction of this thesis with a more detailed problem description and regards to the previously stated challenges. Through this, hypotheses are defined for each research question to build a concept that can provide appropriate approaches for the implementation of mechanisms and therefore means for the evaluation of said hypotheses. While the questions about the visual representation should guide the implementation, the focus of this thesis should remain primarily on building a model for the data flow-oriented coordination of VATs.

Covering all of these objectives would result in a system that allows users to integrate what is necessary for the task at hand and couple what is possible considering technical limitations.

3.2.1 Which principal types of data exchange between VATs exist?

As described in Subsection 2.2.3, there are many different ways in which VATs can acquire data and output results. Whether it is via the local file system, central or federated databases, through the clipboard or rather specific methods such as in the Document Object Model (DOM) of a website. Most typically, VATs are implemented as individual software components (often termed operators) [CR98; HA06] that require specific exchange mechanisms, while the need for a more structured and reusable form of data exchange between independent applications is echoed throughout different communities [BB12; Bau+16]. This heterogeneity is the first and foremost obstacle to overcome for a scalable and open data level coordination. Hence, this research question should be answered by providing means to identify data coordination requirements and limitations for existing VATs independent from the underlying implementation.

Hypothesis: Principle types of data exchange can be defined for various VATs independent of the implementation-specific mechanisms For solving the missing coordination due to the heterogeneity of VATs, one should develop a typology that abstracts from the many different technical realizations. This should be achieved by grouping them according to their characteristics, e.g., data exchange should be unidirectional, bidirectional, synchronous, or asynchronous. It can be assumed that, out of all possible type combinations, only a few are actually relevant for the specific scenario.

For example, parallel coordination and sequential coordination of VATs usually require different types of data coordination mechanisms to realize their coordination needs. These coordination types should be selected to define links between existing VATs so that their principal behavior of data exchange can be queried from a joined structure to route data from one VAT to another via a path that exhibits the necessary behavior without detailing the underlying technicalities.

3.2.2 What parts of the data can or should be exchanged between VATs?

As seen in Subsection 2.2.3, common data management mechanics are often concerned with optimizing performance to enable faster computations and real-time interactions with the visual part of a VAT. It is important to maintain the scalability of the coordinated VATs (especially with larger datasets, as they are more often the starting point for the necessity of visual data analysis). Passing entire datasets repeatedly between pairwise coupled VATs can therefore be assumed to be wasteful, if more often only small subsets of interesting data may be sufficient. However, due to the independency of VATs, information about the internal structure can not be inspected or even adjusted for every included VAT. This leaves only the transition between VATs to exchange salient parts of the data.

Hypothesis: Scalable data exchange can be achieved by relying on transitional input and output management For solving the scalability of data in VAT coordination, one should establish a fine-grained data synchronization concept that takes coordination requests together with temporal and capacity constraints as an input and defines a suitable data flow on top of the underlying structure as an output. This flexible coordination structure should allow users to integrate VATs and data sources at different times during the process through minimal invasive coupling. Pairwise connections should thereby be based on the input and output of tools to produce, consume, and store varying amounts of data that have undergone changes since the last data exchange, are currently visible or will become important for the next step in the analytic workflow. Such saliency factors should be investigated, and data subsets should be derived accordingly based on the coordination types that are clarified in Subsection 3.2.1. For example, it is reasonable to assume that changes to the currently visible data should be reflected as quickly as possible in other VATs and, consequently, require a fast synchronous coordination mechanism. In contrast, data items that will only become important in the next workflow step can be exchanged asynchronously in the background to be shown whenever ready. Together, the selection of salient data subsets and their mapping onto the available coordination types define the data flow between the VATs.

3.2.3 How does someone annotate the data flow to provide meaningful information?

Once the base functionality of data level coordination is present through the dataflow, it might be helpful to not only pass raw data but also capture supplementary information. This output could be either produced by the user (e.g., through manually entered remarks), the VATs (e.g., through automatic screenshots), or process knowledge (e.g., through global parameter settings) together with the processed data. Once the output is detected, the data can be annotated with additional information during its transition from one VAT to another, extending the concept of produced and consumed outputs to include information produced by the users, the individual VATs, or the coordination framework itself and consumed by any of those three entities.

Hypothesis: The analysis process benefits from supervised annotation mechanics To optimize inter-process human-computer communication, one should define to what extent automated annotations can be used and at what point user-generated annotations are required. The result should be a fine-grained data synchronization concept that takes coordination requests together with temporal and capacity constraints as an input and defines a suitable data flow on top of the underlying structure as an output. This way, users can not only define the transfer of raw data, but also user comments, parameter choices, and process knowledge to other users, tools, and the overarching unified user interface.

3.2.4 What should be shown to represent the user's workflow?

As described in Subsection 2.2.5, there are various methods for VATs to display and organize analytical information on screen. The contents to be shown include panels and views of the current workflow step, as well as information about available data and previously applied or successive tools. Each of these solutions is typically driven by a specific application scenario with a known subject domain and technical limitations. In order to provide a rather independent solution, the limited screen space is therefore the first obstacle to overcome in order to present information through a unified UI at a given point in time. Moreover, during the analysis, users might come up with new insights, which raise new questions and require further tools to provide additional data, views, and functionality. Based on these adaptations, an update of the currently selected UI components might be required.

Hypothesis: A unified UI can be constructed independent from the amount of VATs that are to be shown on screen For managing an unknown amount of VATs on the limited screen space, one should develop a concept for unified UI that must take into account the information that needs to be displayed at a given time and the context in which it is displayed. Given the limited visual attention of human beings, it also needs to be carefully considered how much information should be displayed. Relevant information should therefore be extracted from an underlying structure to let the user

interactively reduce or extend the selected parts of the visual process representation. The result of this procedure is a set of UI components that should be displayed in fine-grained distinction.

3.2.5 How can the workflow be represented in a meaningful way?

Showing the information of the analytical process on screen (see Subsection 3.2.4) is only the first part of the visual task in VAT coordination. In regards to the user's orientation, the layout of the UI components should follow their intended use, while supporting the user in his task. That means that the user should not be confronted with a new UI layout every time due to UI components fluctuating during an analysis session or between multiple sessions. This is especially important for the layout and adaptation of the current tools, the display of information about preceding and successive workflow steps as well as the overview on the general coordination structure.

Hypothesis: There are orientation rules for a unified UI that can be defined without prior knowledge about the final implementation To ensure the orientation throughout the visual analytics process, one should explore various options for realizing smooth transitions between UI configurations and provide navigational features putting the user in control of the switching. This could be achieved by displaying information about current, preceding and successive workflow steps as well as deciding about a permanent or transient solution for the underlying coordination structure that should enable the analyst to actively manage available, include new or remove VATs. This encompasses suitable visual controls to trigger data flow changes, and semi-automatic feedback strategies on these changes in the workflow as well as options to scale-up views on demand to make space for redecorating them with additional labels, or otherwise to abstract them to give more space to a particular view of interest.

3.2.6 How can the user interactively control the UI ensembles?

Global parameter control can be beneficial in many cases. An example of this would be settings such as color palettes or transfer functions that should be consistent across tools to minimize the cognitive load of the user. However, under certain conditions, localized parameter settings may be preferred. A unified UI should provide options for both local and global parameter control. This requires ways for gathering and organizing parameters from individual VATs to balance them with interactive control mechanisms of global settings. Such adjustment and re-parametrization facilities should replace tedious and repetitive manual operations while maintaining the "freedom of the user" to make local parameter choices in individual VATs without overwriting them through global parameter changes.

Hypothesis: There are options for interactively controlling local and global parameters during the analysis that would benefit tool coordination For increased

effectiveness when steering the display, one should supply global control and user support. A combined visualization of workflows, tools, and data would make it possible to view all relevant information at any time. Beyond that, the unified UI should take appropriate means for interactive control of the UI ensembles.

Addressing the research questions raised, it is more beneficial to explore a range of diverse approaches, potentially yielding multiple solutions. As a matter of fact, there may be no ultimate solution covering all aspects of VAT coordination through the proposed unification interface. The main question of this thesis should, therefore, be: "Is there a way to coordinate multiple visual analytics approaches through a uniform interface so that the known user workflow of a domain expert can be optimized?".

3.3 Predetermine Requirements and Limitations of the Approach

Based on the analyzed challenges and the described research questions, this section defines the requirements and limitations for the methods and their prototypical implementations. Through collaboration with different user groups, several design requirements for a unified user interface for coupling VAT tools could be identified in collaborative work with Fraunhofer IGD and Aarhus University [ASU13; Sch+19b; Röh+23].

Independence of Domains: As already discussed, a high degree of independence from application domains is the most important requirement. This means that the intended coordination interface does not fall back on existing technical architectures of VATs to change them but rather exploits coordination possibilities apart from these individualized implementations and uses them for itself. Therefore, methods for the categorization of data exchange between VATs need to be defined in order to make meaningful adjustments.

Access to Tools: In a workflow, the temporal order of VATs and their interplay is typically only implicitly defined. On the one hand, this means that users have to remember which VAT to activate for the current work step. On the other hand, they also need to determine what content to display at the same time with these tools. The contents include currently required panels and views, but also information about available data and information about already used or subsequent tools. Therefore, access to the right tool at the right time must be supported.

Connection of Tools and Data: In general, the entire data is not passed between VATs over and over again during the execution of a workflow. Instead, the data are incrementally reduced and sometimes even selectively extended to extract valuable information. Thus, users must not only supply the original data to the tools at the beginning of the workflow, but also manage the exchange of various data pieces between the tools during subsequent steps. This includes taking care of all necessary data transformations between the tools and updates if parts of the data

have changed during the analysis. Supporting access to the right data at the right time and establishing a link between the data and the tools is thus a requirement.

Arrangement of Views: When activating different tools, the layout of their views should correspond to their intended use in the respective part of the workflow. If the tools are used one after the other, their views may simply be interchanged. However, if they are used in parallel, their views must be organized on the screen in a meaningful way so that they can be used together effectively. In addition, navigational features are needed that not only give the user control over switching between workflow steps and associated view layouts but also help to maintain orientation at the same time. Consequently, support for obtaining the right view layout at the right time is needed to manage the visual output of tools.

Interactive Control over Tools and Data: On a grand scale, visual analytics solutions provide technology that combines the strengths of human and electronic data processing [Kei+08]. Visualization serves as the medium for a semi-automated analytical process, where humans and machines collaborate by leveraging their unique capabilities for the most effective results. Hence, the user has to be the ultimate authority in directing the analysis, while the system has to provide effective means of interaction to focus on their specific task. This is also partly due to the independence of domains, as workflows might need to be adapted by experts due to new or changed domain knowledge. Imagine a scenario in which an analyst is investigating a hypothesis based on a dataset they have been examining for several days, only to realize that there are still potential outliers or errors in the data. Current VATs do not make it easy to non-destructively evaluate how different options such as removing, transforming, or retaining such data points affects previous analysis paths and their results. To explore multiple alternative analysis tasks, users often find themselves needing to redo portions of their analysis or backtrack through earlier interactions to pursue different hypotheses. This means that the information or sequence of tasks and therefore the order and combination of data sources and VATs have to be interactively configurable and not be pre-defined.

Overview and Steering of the Visual Analysis Process: After configuring the workflow according to the needs of the one's application domain, the domain experts should be in charge of steering the visual analysis process. They should be able to decide weather they can traverse the workflow in a sequential order to jump between different tasks in case that some interesting insight might be gained from re-iterating previous steps or skipping some optional transformations. Therefore, the domain user needs some sort of overview and steering mechanism that enables them to explore the existing workflow without directly re-configuring of what is identified as a current "gold standard workflow".

In order to limit the technical problem of VAT independence for the concept of this theses, the following subsections make assumptions about the analysis process that are consistent with these described requirements.

3.3.1 Definition of Visual Analytics Tools

Visual analytics tools form the basis for implementing a coordination model as they are simultaneously the driving force of interactive user-driven analysis. Hence, in order to understand how user workflows can be coordinated through a overarching system, a definition for tools has to be clarified first. However, as seen in Subsection 2.1.4, the interdisciplinary origin of visual analytics causes various terms to be looked at from different perspectives. A tool may range from a simple script to a fully fledged framework in terms of software development, while it could also mandate specific devices for user interaction in the field of interaction design. In its most rudimentary form, any visual analytics tool should include the following features that are mandatory for user-driven data analysis:

Data Exchange Interface: A VAT must have an available interface that allows users to feed in data and document results throughout or at the end of an analysis. It can be assumed that each VAT has some sort of data input and output as the least common denominator for the exchange.

Computational Core: A VAT must have an analytical component that not only visualizes the data but also enables external intervention (e.g., through user interaction).

Visual Information Display: A VAT must have a visual component that displays the read data and allows manipulation of the display for user interaction.

This fundamental assumption already narrows down the scope of what a VATs is, as purely computational tools or representative tools do not fall into this category. Making this distinction is necessary to account for the highly variable degree of automation in the knowledge discovery process. Computational tools can often be integrated well to tunnel intermediate results since they can perform pre-defined operations automatically. This eliminates the need for user interaction in most cases, since the end result of the process provides the most important outcome for gaining knowledge. Representative tools, on the other hand, only serve the user to understand the underlying data set. The knowledge gain in this case arises from the manual interpretation by the user and can therefore only be reused though tedious documentation of intermediate results or not at all. Visual Analytics Tools, however, carry both computational and visual components, which should allow for the semi-automatic analysis in pre-defined process chains.

3.3.2 Definition of User Roles

As shown in Section 2.1, the role image of a visual analytics user has changed over time. While in the early 2000s "users" were mostly the visualization experts, they are now more often the domain expert that work with the created tools. These domain experts usually have an overview of their application area, but it cannot be assumed from the outset that they also have the corresponding technological knowledge to link different tools together. Various approaches have been suggested to streamline application

development, ranging from high-level programming languages like Ruby and Python to visual programming tools such as Visual Basic. Within the last decade, there have been a number of proposals for tools aimed at non-programmers, including WYSIWYG tools that assist in the development of data-driven web applications [Yan+08] or visual interfaces through which users can assemble pipelines for building web mashups [Pru07]. More often, the development of cross-tool VAT use cases involves more than one party due to the complexity of tasks and their interdisciplinary nature. For the scope of this thesis, this is reflected in different user roles for the conceptual approach, as users with different levels of technical or domain-specific knowledge have different requirements for interacting with a system. Considering the scope of the evaluation within this thesis, these user roles have been simplified to two parties [Non+22].

The *domain experts* have knowledge of the domain workflow and the associated sequence in which the tools must be applied. They are the ones to apply the workflow by performing actual data analysis using the given tools to draw conclusions based on their professional expertise. In terms of analytical tasks, they constitute as the central element in the deployment of processes as only they have enough background knowledge and experience to interpret the data at hand in order to extract new information from the displayed material.

The *technical experts* or visualization authors possess knowledge about data management, data exchange, data visualization, and available inter-process communication methods. They are responsible for coordinating the VATs and configuring the data exchange based on a given domain workflow.

Both roles are meant to work together to establish how the available VATs can be combined and what options are feasible for the task at hand. Hence, the provided concept of the thesis should assist both the technical expert in configuring a suitable toolchain as well as the domain expert in navigating it and executing the workflow accordingly. Rather than fully automating the whole knowledge discovery process (as often tried within other computer science subjects), this cooperation of role models is meant to explore the realm of possibilities together and detect fundamental problems in existing approaches. However, to ease the configuration process, starting, stopping, and switching between VATs at execution time should be automated as much as possible.

4 A multifaceted Model for the Orchestration of Visual Analysis Tools based on the Separation of Concerns

This chapter provides a novel multifaceted model for the coordination of independent VATs in pairwise toolchains. Based on the previously established challenges and requirements in Chapter 3, the model employs a layered approach that aims to follow the flow-based nature of the visual analysis process. It thereby follows its interactive behavior by distributing tools, data and parameters based on temporal, spatial and conceptual separations towards different sources, user roles and user interface ensembles to integrate what is necessary and couple what is possible.

4.1 Flow-oriented Structuring of the Analysis Process

As discussed in Section 2.2, the visualization during an analysis is a repetitive process that requires data to unfold the stored information from various sources. Since every part of the process (i.e., data and view) is based on the user's workflow, it can be considered flow-oriented. Following the previous distinction (see Section 2.2.4), the required VATs have to therefore coordinate data, function, and view based on the flow-oriented behavior to find solutions for the conceptual, temporal, and spatial separation in order to employ interactive knowledge discovery.

4.1.1 Identifying relevant Entries for cross-tool Coordination

Following Munzner's nested model for visualization design [Mun15], there are four different stages for information visualization that also apply to the visual analysis of unknown data.

1. *Domain Problem and Data Characterization:* At the beginning, one needs to characterize the problems and data in the vocabulary of the particular problem domain.

2. *Operation and Data Type Abstraction:* Next, the identified problem needs to be mapped into abstract operations and data types.

3. *Visual Encoding and Interaction Design:* These operations need to be supported by the design of visual encoding and interaction matching with the abstract representation.

4. *Algorithm Design:* Finally, algorithms are created to execute visual encoding and interaction designs automatically and efficiently.

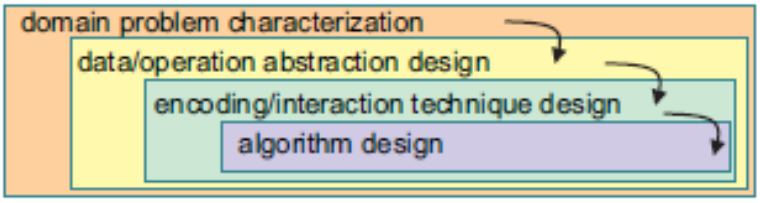

Figure 4.1: Nested model of visualization creation by Munzner [Mun09]

The output from each level above is input to the level below (see Figure 4.1), bringing attention to the design challenge that an upstream error inevitably cascades to all downstream levels. This provides prescriptive guidance for selecting suitable evaluation methods by pinpointing validity threats that are specific to each level.

As shown in Section 2.2, previous work on coordinating and linking visualization and analysis tools is mainly focused on the implementation and software engineering aspects, which resembles the *algorithmic level* in terms of Munzner's four levels of visualization design (see Figure 4.1). With the lightweight coordination approach [Sch+19b], the remaining three levels in the nested visualization design model are discussed to identify relevant entries for coordinating independent VATs.

The situational level aims to understand the application domain, the application data, and questions, as well as the analytical processes that derive from them. This can be captured by providing information about the used toolset in a domain, as well as common orders of use for domain-specific analyses. This is defined as the *usage flow* that determines which VATs are used in which combination – i.e., subsequent or concurrent mode of operation.

The abstraction level aims to understand the data transfer in terms of which data in which format is exchanged as a result of which transformation. This can be captured by providing information about the data exchange between the independent VATs, as well as provenience information on the analysis process itself. This is defined as the *data flow* that determines what the VATs are working on and what they produce – i.e., their inputs and outputs.

The encoding and interaction level aims to understand how the data is visually displayed and how this display can be adapted. This can be captured by providing information about the parameter settings that govern the display, that can be changed to adapt the display, and that can be passed along for consistency of

displays among tools. This is defined as the *control flow* that determines how the tools are used – i.e., the steering of their inner workings.

The defined structure of usage flow, data flow, and control flow is further justified by the previously examined state of the art as this abstraction includes the different approaches for data management, data analysis, and visualization in Section 2.2.

4.1.2 Establishing Minimal Invasive Entry Points for Visual Analytics Tool Coordination

While the classification by situational, abstraction and interaction level helps to reflect the flow-oriented nature of visual analysis onto the process, the orchestration of independent tools constitutes for the primary obstacle. In order to enable coordination for individual VATs on each of these levels, the coordination methods have to be minimally invasive based on the previously defined requirements (see Section 3.3). VATs are therefore seen as *"black boxes"* with no assumptions about the internal structure or mode of operation so that the available entry points are for the most given by the specific domain workflow [Sch+19b]. Nevertheless, it can be assumed that the following ports exist for every VAT, regardless of whether they are actually provided by the tool itself or by another entity, such as the operating system [Sch+20].

Start/Stop Mechanic: Starting and stopping a tool, along with receiving notifications about its initiation or termination, constitutes the most basic capability of any tool. The invocation and termination of a tool are therefore modeled via the port *Start/Stop* and the respective notifications via the port *Started/Stopped*.

Data (in/out) Mechanic: A VAT requires input data on which to conduct its analysis. The passed input may yield results that the tools pass back in return. It is common for most VATs to have some ways of loading data, which can be modeled as a port $Data_{IN}$ and saving results via the port $Data_{OUT}$. Note that the term data subsumes numerical data as well as image data.

Parameters (in/out) Mechanic: A VAT's behavior is generally parameterizable. If parameters can be passed to the tool – such as command line options during invocation – this is represented via the port $Param_{IN}$. If parameter settings can also be stored away for later re-use – such as in a config file – this can be captured through the port $Param_{OUT}$.

Events (in/out): VATs may permit a certain degree of steering and observing its inner workings. Commands may be passed to a VAT through the port $Events_{IN}$ – in order to run a certain computation – and the progress and success of carrying them out may be passed to the outside world via the port $Events_{OUT}$.

Visualization & Interaction Mechanic: In most cases, VATs generate visualizations of the input data and display them to the domain expert, who can then make interactive

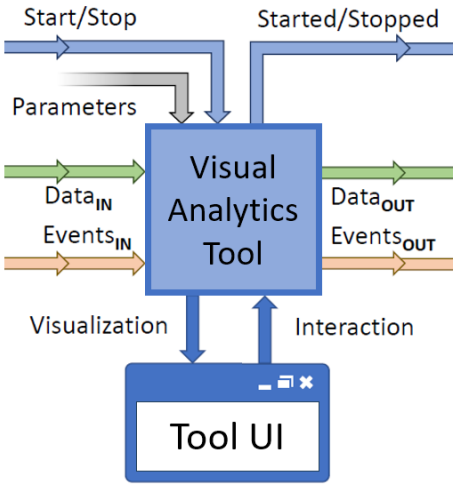

Figure 4.2: Visual representation of minimal invasive entry points for an unspecified VAT. The core of the VAT acts thereby as an interface for the in- and outputs of data, event, parameter, visualization, and interaction ports.

adjustments to them. These aspects are captured by the ports *Visualization* and *Interaction*.

An unspecified VAT would therefore look similar to the result of the abstract representation shown in Figure 4.2 that portraits the core of the VAT as an central interface for the previously defined ports that interact on it. It should be noted that VATs do not necessarily control all of these input and output capabilities. In certain instances, it might be possible to circumvent these limitations – such as inferring otherwise inaccessible parameters from a tool (i.e., $Param_{OUT}$) based on the updated results (i.e., $Data_{OUT}$), for example, by extracting color scales from visualizations [PMH18]. In other cases, it might be necessary that the technical expert steps in, e.g., by manually setting parameters or choosing dedicated presets.

4.1.3 Modeling pairwise Connection of Visual Analytics Tools based on lightweight Tool Coupling

By assuming that VATs follow the previously established definition including minimal invasive entry points (see Subsection 4.1.2), we have been able to define "lightweight coordination" of VATs based on the underlying flow-oriented structure [Sch+19b]. The approach is *lightweight* in the sense that its realization is considered to be pairwise between tools. Instead of having to provide an all-encompassing framework that coordinates between all possible VATs and anticipates all possible combinations in a top-down manner, a bottom-up approach is used that works by coordinating only between the tools and levels necessary for the task at hand. This allows for a more custom utilization of different coordination mechanisms between tools depending on the available VAT interfaces.

The Usage Flow At the lowest level of coordination, there is the temporal order of VATs according to existing analysis procedures in a domain. This is defined through the domain experts workflow including suitable tools for the given tasks and a sequences in which they should be used. In the lightweight model, this sequence is described as the *usage flow* capturing a succession of VATs, as these are used by the domain expert in pursuit of a particular analysis goal. This type of coordination is achieved through a bilateral connection among two tools indicating their *coordination order* (see Figure 4.3) by starting one VAT subsequently or concurrently, depending on another VAT having been Started/Stopped:

- Subsequently: $(Tool_1.Stopped) \Longrightarrow (Tool_2.Start)$
- Concurrently: $(Tool_1.Started) \Longrightarrow (Tool_2.Start)$

VAT coordination modeled and realized at this level already provides as much as an automated guidance of the user along a predefined path of analysis, in the spirit of approaches such as *Stack'n'Flip* [Str+12]. While the domain expert still has to handle processes such as moving data back and forth between the tools, or parametrizing their visualizations to match up manually, the coordination order between VATs allows to automatically move forward or backward during the analysis. In addition to enhancing user convenience, this also guarantees comparability across different tasks, as VATs are always invoked in line with the predefined usage flow. This is important to ensure that no VAT is left out by mistake or in cases where carrying out tasks in different orders yields different results, for example, when performing dimension reduction [KRM18] and clustering [Wen19]. Furthermore, it helps prevent the accidental omission of any VAT, such as forgetting to normalize the data before processing it.

The Data Flow The next level of coordination is about getting data in and out of a VAT. Besides starting and stopping, this is arguably the most important aspect of a VAT, as without data, there is nothing to work with. The data flow captures how data is passed into VATs, transformed into analysis results, and passed on to the next VAT in line. In the lightweight model, this coordination is achieved via *coordination channels* (see Figure 4.3) that capture the data exchange from a VAT's output to another VAT's input, as well as any data transformations f_D along the way:

$$Tool_1.Data_{OUT} \xrightarrow{f_D} Tool_2.Data_{IN}$$

Note that "passing data" can be a complex problem and research on data integration and scientific workflows has established various approaches to do so [Lud+06]. For modeling the coordination on the data level, it is enough to know that one of these approaches underlies a channel and realizes the data transport. It is up to the domain expert to determine how to handle the data within the currently utilized VAT, such as interactively setting parameters for computations and visualizations. However, having input data and results from previously used VATs delivered automatically to the current VAT takes

care of any tedious conversion between different data formats, competing standards, or sometimes just different versions or interpretations of the same standard [Sch+17]. It also ensures that the domain expert is not working on stale data, as the coordination channels always deliver the most current data. Therefore, loading an old dataset can only occur intentionally and not inadvertently. Coordination channels can further be utilized to store a snapshot of the last data handled by them so that when revisiting a previously used VAT, its last input data is still accessible, thus providing a coherent analysis experience forward *and* backward along the usage flow.

The Control Flow The last level of coordination resembles the most common understanding through interactive controls, where, for example, filter operations, selections, or adaptations of a color scale in one VAT would also affect other VATs. If the corresponding view and data parameters are captured and accessible, coordination on this level can be used to provide any of the common coordination patterns, such as linking & brushing, navigational slaving, or visual linking [Wal+10]. How exactly these affect subsequently or concurrently used VATs is subject to the concrete nature of the coordination. In the lightweight model, this coordination is achieved via *coordination rules* (see Figure 4.3) that capture not only the mere passing of parameters between VATs but also the "coordination logic" behind the linking they realize. In cases where this linking is supposed to have changes made in one VAT being reflected likewise in another VAT, the coordination rule relays the corresponding parameters as they are. Yet, in cases where this linking is intended to have a complementing effect (for instance, when the other VAT does not display the selection but rather everything that was unselected), the coordination rule relays the inverse of the corresponding parameters. Coordination rules are modeled as functions f_P with the identity $f_P = id$ as a default:

$$Tool_1.Param_{OUT} \xrightarrow{f_P} Tool_2.Param_{IN}$$

As it was the case for coordination channels, the domain expert must still manage and steer the currently used VAT. However, if it is applicable to the task at hand, this steering is picked up on and automatically mirrored or complemented in other VATs. This encompasses not only VATs that are utilized concurrently but also those employed subsequently, as interactive selections or carefully tuned color scales can be passed on as parameters to the next VAT. Through this, the approach guarantees consistent settings but further relieves the domain expert from having to make the same interactive adjustments multiple times for each VAT.

Flow-oriented Lightweight Coordination

Taken together, the lightweight model shown in Figure 4.3 defines the pairwise exchange between two VATs using:

- a *coordination order* describing any automation of the usage flow,
- a *coordination channel* describing any transmission of the data flow and

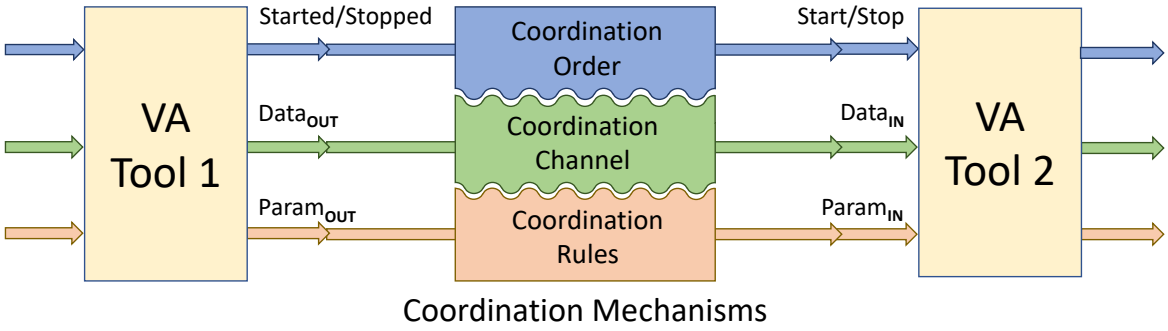

Figure 4.3: Visual representation for the conceptual abstraction of lightweight coordination between two VATs [Sch+20]. The *coordination order* models any temporal dependency between two tools (i.e., their subsequent or concurrent use). The *coordination channels* capture ways to exchange data between tools, including any necessary data transformations along the way. The *coordination rules* describe automated syncing features between tools as action/reaction pairs.

- a *coordination rule* describing the synchronization of the control flow.

It can thus be expressed as a 4-tuple $cm = (tools, usageflow, data\ flow, control\ flow)$ consisting of the following sets:

- the set of *tools* as it is given by the analysis setup
- the *usageflow*, defined as the set of all coordination orders $(source, [started|stopped], target, [start|stop])$ capturing the execution sequence between a source and a target tool
- the *data flow*, defined as the set of all coordination channels $(source, target, f_d : data_{out} \mapsto data_{in})$ capturing data transfer and transformation (f_d) between a source and a target tool
- the *control flow*, defined as the set of all coordination rules $(source, target, f_p : param_{out} \mapsto param_{in})$ capturing parameter exchange and modification (f_p) between a source and a target tool

This initial approach [Sch+19b] already represents the basic idea of flow-oriented process coordination and provides entry points as a first step in the direction of interconnected VAT coordination in the context of the previously defined conceptual separation challenge (see Subsection 3.1.2).

4.2 Characterization of Data Exchange

After establishing a pairwise coordination model, it is necessary to go beyond simple entry points for coordination and instead towards identifying connection points for the data

exchange. To this cause, a taxonomy that captures the range of data exchange possibilities through a detailed description of individual characteristics was developed [Non+20]. This characterization model can be used to classify systems for the coordination of mostly independent VATs.

4.2.1 Examining the State of pairwise Data Exchange

Generally speaking, data exchange can be understood as a process of sending data that is structured under an existing source schema and converting it into a format that adheres to a target schema [DHI12]. This is similar to the related concept of data integration (see Section 2.2.3) except that the data is intentionally restructured (with a potential loss of content), which might result in non-transformative instances given all of the constraints. Using the example of data exchange between pairwise coupled VATs, this would consequently be the schemata of the sending VAT's $Data_{OUT}$ that is transported and potentially transformed by the receiving VAT's $Data_{IN}$. The underlying schema of $Data_{OUT}$ and $Data_{In}$ could, for example, also be an interchange format within a single application domain. In this case, the required mappings are then created to indirectly transform or translate each specific source schema into each specific target schema using the interchange format as an intermediate step [Are14]. However, due to the previously defined independence requirement (see Section 3.3), this approach is not viable anymore since there is no interchange format that can transform *any* existing data type.

4.2.2 Finding Ways to Characterize Data Exchange

Through our research, we found that data exchange needs a fitting characterization of different aspects [Non+20] to not only define entry points for transformation but also handle the possibilities and limitations of the information transport. While previous methods focused solely on the transformation part of diverging data formats, we focused on finding suitable mappings for data management processes overall. For this purpose, a model was developed based on the *Five W's Model* by Zhang [ZH13] that asks for different conceptual aspects through five questions of *What?*, *Why?*, *Where?*, *When?*, and *Who?*.

Nevertheless, the *Who?* in this model was adjusted into a *Which?*, as this approach is talking about the communication between two VATs instead of real users. This leaves five questions for the model shown in Figure 4.4 that characterize different aspects of the data exchange process between individual VATs.

Data Characteristics

Before delving into the exchange process, one should first clarify the degree to which unknown data sources can be predefined. Depending on where in the visualization pipeline a VAT operates, data can refer to raw data, prepared data, focus data, geometric data, or image data [dB04; Tom+06]. As Schulz et al. [Sch+17] describe it, data characteristics are often assessed in an initial data profiling step, including, for example, statistical

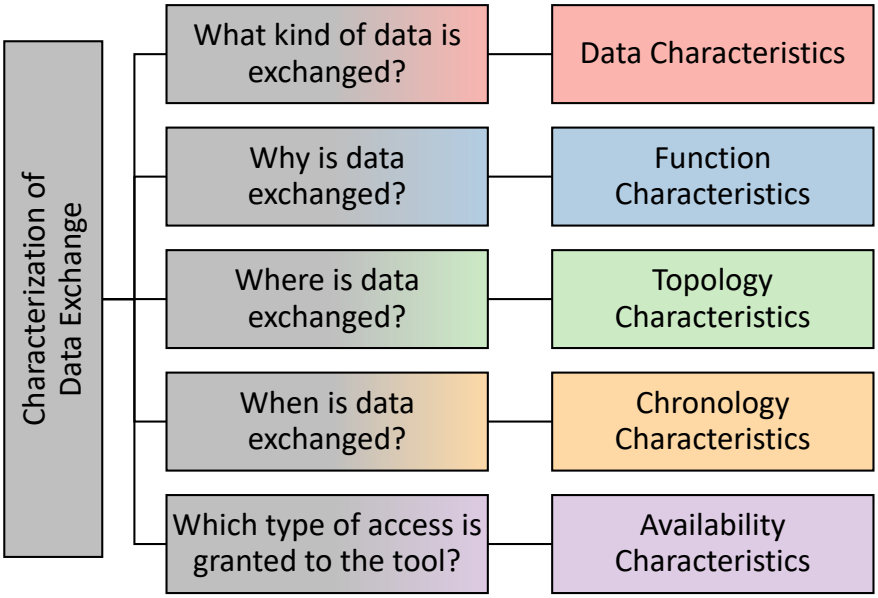

Figure 4.4: Visual representation of the five characterizing questions for data exchange between independent VATs [Non+20].

properties of the data and information on the data's well-formedness, which can be used during the subsequent analysis to adequately parametrize views and to highlight or exclude data items [Sch+16]. Although the term "data" is very versatile and can involve various extents depending on the discipline, it is generally understood as a set of values from which information can be determined. To define this set of data more precisely, one must consider the following questions:

- *What type of information can be extracted from a data source?*
- *What structure is used to organize and store the information?*
- *What partition of information is necessary for the data exchange?.*

Informational Contents Data from the visual analysis domain usually refers to values that can provide insight into a specific usage scenario, such as n-dimensional vector fields, 3D geometry, or spatial sensor data. For the analysis process itself, however, other information, such as records of interactions or usage behavior, can also be purposeful for knowledge discovery. Based on this, two aspects of the information content of data can be defined to distinguish between the data set itself and the overlaying data descriptors.

A *data set* is a collection of values, items, or facts that are specified by primitives (for example, numbers, strings, or vectors) based on predefined value scopes. This refers to most information, which is stored in simple file structures that either hold raw data, prepared data, focus data, geometric data, or image data. Comprehensive data analysis methodologies start with an initial analysis [Ade08] or data profiling [Gsc+14] to assess

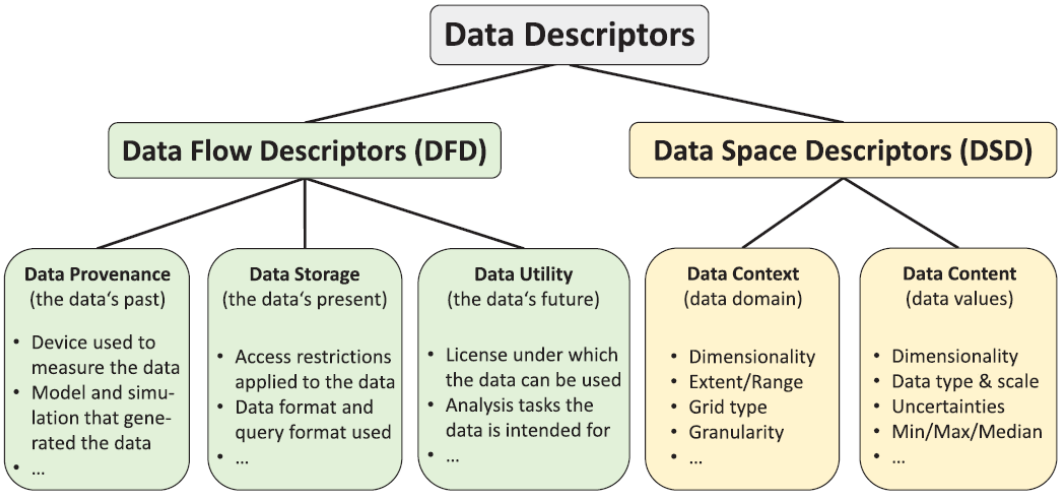

Figure 4.5: Classification of data descriptors from Schulz et al. [Sch+17] arranged from general (top) to specific (bottom) with added examples for common descriptors.

this type of information. However, input datasets rarely carry information about aspects beyond the raw data. Hence, more descriptive representations of datasets are used today.

A *data descriptor*, on the other side, encapsulates basic information about the data set, such as provenance information, storage schemata, uncertainties, or general metadata in order to parameterize views, highlight or exclude data items [Sch+17]. An overview on possible data descriptors is shown in Figure 4.5. This type of information can be used by the overarching system during the subsequent analysis to automatically recognize possible data transformations, categorize similar VATs, or adequately parametrize views and highlight or exclude data items. Examples of such metadata are semantic information through tags or keywords, such as HTML or CSS elements, or editorial information, such as date, time, and authorship in stored documents [CYM11].

Data Formats Over the past years, countless data formats have been proposed and even today, new file formats are still being created in order to provide application-specific properties. Combining data sources independent of their origin is a challenge, which is discussed in various research papers [Lo +19; MM04a] and patent applications [Jud+04; Mül18]. To understand the fundamental differences in data from an abstract perspective, it is common to categorize data based on their structuredness. Examples of this are shown in Figure 4.6.

Structured Data refers to organized information that follows a predefined data model (also called schema) [Bar+13]. These types of data formats are typically found in relational database management systems (RDBMS), where attributes are stored in specific columns of a table [SRJ18]. Common examples of structured data include reports, logs, tags, or objects in large repositories, such as payroll, inventory, or account management.

Unstructured Data is essentially the opposite as it includes data with no predefined

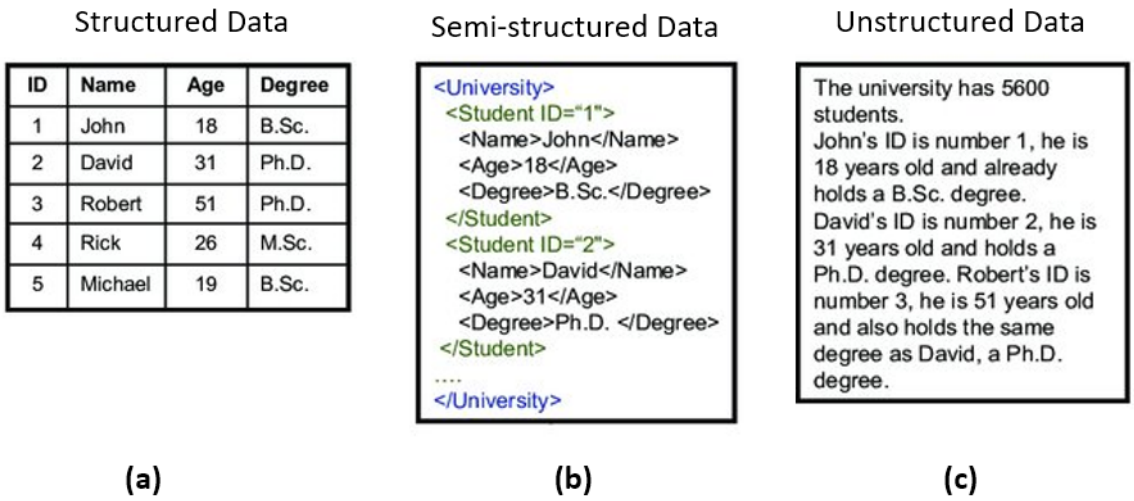

Figure 4.6: Examples for structured (a), semi-structured (b) and unstructured (c) data.

schema [Bar+13]. Unstructured data originates from machine- or human-generated information and is categorized into two common types [SRJ18]: *Non-textual unstructured data* are multimedia files such as images, sounds, or videos, while *Textual unstructured data* define readable files such as email messages, office documents, or metadata.

Semi-structured Data contains predefined structural elements, such as tags or markers, combined with free-form, unstructured components. This helps to separate semantic elements and establish hierarchies of records and fields within the corresponding information [SPM04; ABS99]. Therefore, it is also known as schemaless or self-describing structure. Common examples of semi-structured data formats include those based on Extensible Markup Language (XML) or JavaScript Object Notation (JSON).

Quantity of Exchanged Data The previous sections highlighted the diversity of information within a data source. In order to achieve a high-performing and responsive coordination, it may be necessary to partition the number of variables that are exchanged via the systems architecture. This quantity is described using three different characteristics.

A *Full Data Exchange* is the simplest solution for the information transfer, where the entire dataset is exchanged as a whole. A typical example of this are snapshots, which capture the state of a system at a specific point in time [Str+12]. This is particularly useful for maintaining a coherent analysis experience forward and backward along toolchains.

A *Segmented Data Exchange* is designed to enhance the performance of toolchains by splitting data into segments over time. This enables fast and efficient information processing, which is crucial for the progressive refinement of visualizations [Ang+18]. The manner of segmentation is thereby heavily influenced by the data type. For instance, graph data may be divided topologically, while time-based data is typically partitioned into meaningful intervals.

A *Delta Data Exchange* is carried out by exchanging only at the changed part of the data source, focusing on the urgency or priority of operations. Examples of this include small selections of groups or single parameter manipulations in one VAT that trigger immediate updates of the visual representation in another VAT.

Function Characteristics

After understanding the information within the exchanged data, the next task is to examine its function in the exchange process. In this context, it is important to answer the question: *Why is data exchanged?*
To this cause, the common functions in an exchange process between data and tools are divided into four categories as shown in Figure 4.7.

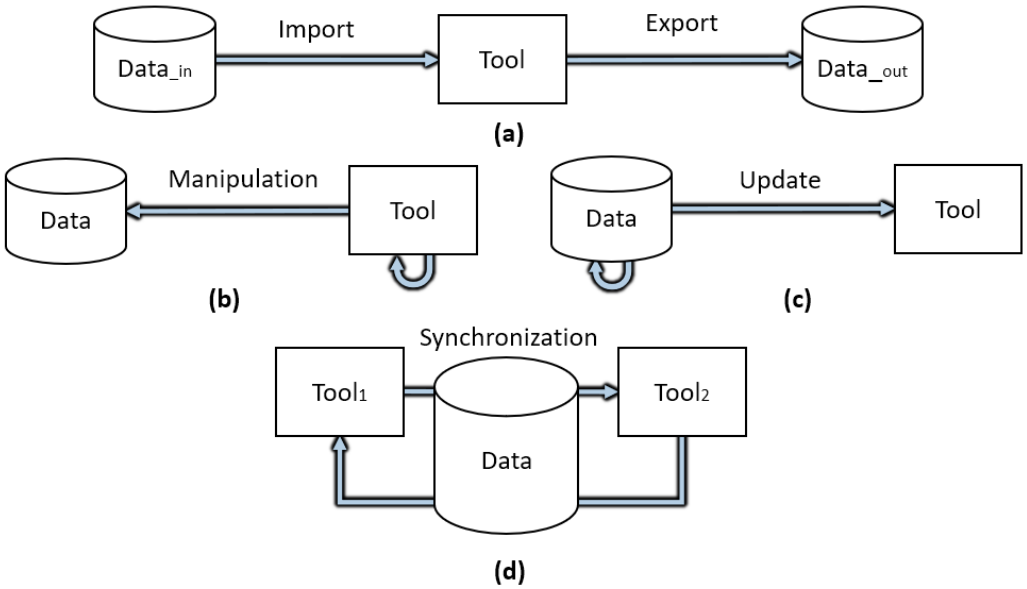

Figure 4.7: Visual representation of the function characteristics with (a) showing the import to and export from a tool, (b) showing the modification of data from a tool, (c) showing the update of data towards a tool and (d) showing the synchronization of data between tools.

The *Import* and *Export* operations on data are the most fundamental operations in the data exchange process, as VATs consume and produce information from data sources. In this case, the full data set would be exchanged in order to be used by the corresponding VAT.

During analysis, it is inevitable that information in a data source will change over time. This *Modification* operation on data is commonly performed by saving the progress of one VAT within a file. Depending on the system's infrastructure and the quantity of information, data can thereby be exchanged via full or segmented data exchange.

Another possible operation is the *Update* operation on information by sending additional data in chunked segments to refresh the visual representation. The transmission rate can

be defined by certain time intervals or triggered by specific action events. A common example of this are streaming services, which perform frequent updates by sending newly generated data over the system's infrastructure to the corresponding VAT [Won+03; CFC17].

Synchronization among tools refers to the process of aligning data and parameters between them – i.e., a change to the data in one of them will be reflected in the other. An example of this would be the linking & brushing mechanism [Wal+10], where selections in one tool trigger a highlighting in another tool. This process is usually realized through a bidirectional delta data exchange.

Topology Characteristics

So far, the data exchange has been considered the communication process. However, to perform the communication, there needs to be some sort of medium. Hence, it is necessary to cover the questions:

- *Where is the data exchanged?*
- *Where does the data come from?*
- *Where is the data going?*

Infrastructure There are multiple software infrastructures available for the data exchange between different tools. However, all of these infrastructures can be divided into two fundamental architectures as shown in Figure 4.8.

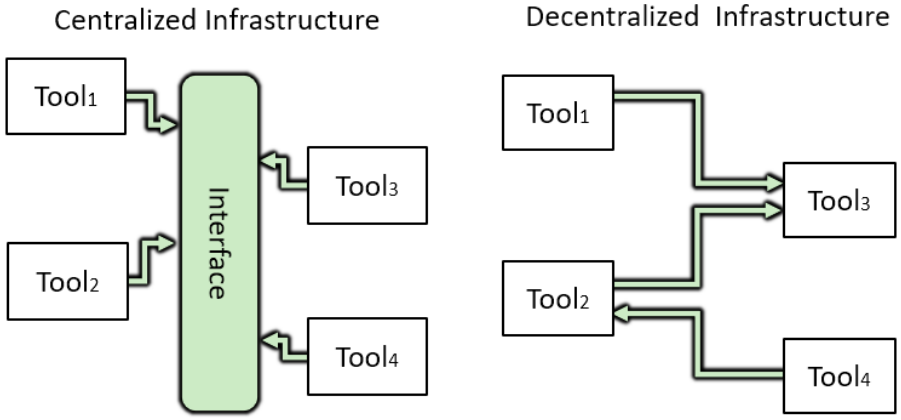

Figure 4.8: Visual representation of both infrastructure models for data exchange. Visual analytics tools exchange data thereby either through a centralized mechanismn (left) or decentralized between each other (right).

A *Centralized Architecture* consists of a single software solution that facilitates data exchange data between multiple VATs through a uniform platform. A common im-

plementation is by using a relational database, which acts as the model in a model-view-controller mechanism. Examples of such centralized data storage are systems like Snap-Together [NS00] or EdiFlow [Ben+11]. Another possibility for a centralized architecture involves utilizing a service bus to broker messages among tools. An example of such service-oriented architectures is the Metadata Mapper [RM11].

A *Decentralized Architecture* provides data exchange through a variety of different mechanisms without a uniform platform. This approach often employs custom connectors to access content from websites or shared network data. Common techniques in this category include web mining and extensive browsing, exemplified by tools like Intertwine [Fou+14], or mashup tools like Mashroom+ [LH14] or VisMashup [San+09].

Relations Until now, VATs are described as some sort of *black boxes* that can handle input and output information [Sch+20]. Following up on this idea, data sources can be either the initial input or the resulting output of a VAT that may be stored in a data format. Given that multiple tools may be involved, various possibilities arise for inputs and outputs of VATs. Consequently, the four key relationships shown in Figure 4.9 have been used to define the relationship between data sources and VATs:

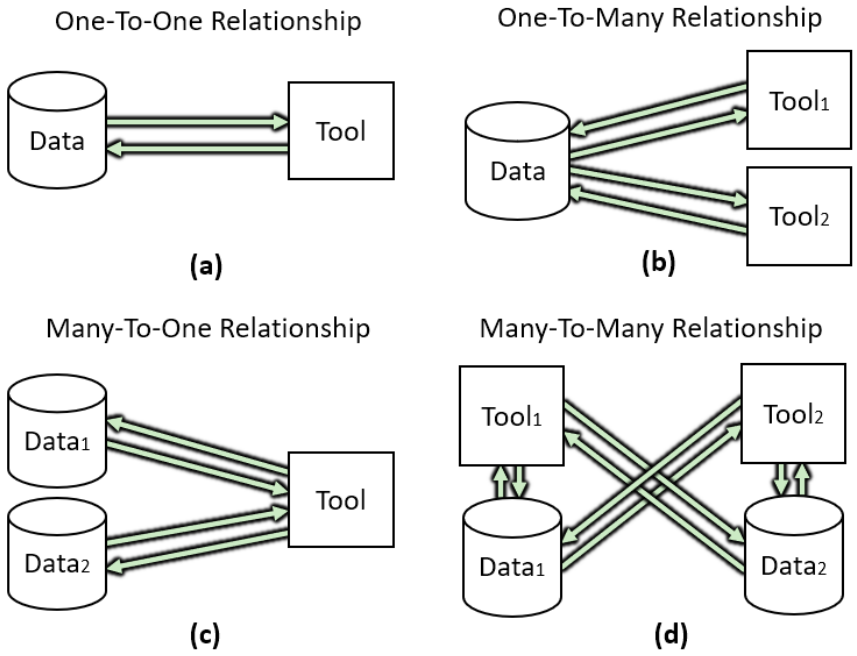

Figure 4.9: Visual representation of the relations for data exchange showing (a) the One-To-One relationship, (b) the One-To-Many relationship, (c) the Many-To-One relationship, and (d) the Many-To-Many relationship that are possible between data sources and visual analytics tools.

The *One-to-One Relationship* provides each tool with exactly one data source. This is the simplest use case, suitable for initializing a VAT or transporting output data between

two subsequently used VATs.

The *One-to-Many Relationship* is applied when multiple data sources are used by a single VAT. To achieve this, data transformation techniques, such as merging between data sets, must be applied to all utilized data sources. The complexity of this problem increases significantly with the multitude of different data formats used for the task at hand.

The *Many-to-One Relationship* refers to one data source that is used by multiple VATs. In this case, Clear communication among the independent VATs regarding the chronological order of operations is needed, as simultaneous access to the same data source can lead to conflicts with multiple versions of the same data source existing at the same time.

The *Many-to-Many Relationship* is essentially a combination of the two previously discussed relationships, where multiple data sources are used by multiple VATs.

Directionality Another important aspect for the data exchange is the direction of the communication process, which is described to be either unilateral or bidirectional.

For the *Unilateral data exchange*, data is transmitted exclusively from one VAT, which is the sender, to another VAT, which is the receiver of information. A simple example for this would be the data exchange between tools with single functionality such as simple command-line tools or data format converters, which are used only once without requiring specific parameter adjustments. Therefore, data is just passed through in one direction of the toolchain.

The *Bilateral data exchange*, on the other side, allows for open communication between VATs in both directions. This corresponds to continuous analysis tasks that involve frequent parameter changes, such as the parallel display or dynamic switches between VATs. Example use cases include data exchanges between multifunctional tools that offer the necessary parameters for comparative or advanced visualization of information.

Chronology Characteristics

As data exchange, especially for big data visualization, is a time-dependent task, it is necessary to schedule the data transfer – an aspect that is often conveniently left out of the discussion [CL10]. Therefore, the following questions need to be answered:

- *When is the information exchange planned?*
- *When is the exchange executed?*

Timeline The timeline for the data exchange process between two Visual Analytics Tools (VATs) can be looked at as a planned execution of different data transfers. A data transfer is thereby the process of passing portions of the data from one VAT to another. This process can be categorized as either synchronous or asynchronous:

A *synchronous data transfer* occurs if both VATs are available for sending and/or receiving data at the same time. A successful synchronous transfer requires, therefore, an open communication channel that remains active during the entire exchange process.

An *asynchronous data transfer* allows for delayed data exchange, where the sending and/or receiving VATs are available at different times. This type of communication is typically established by the system's infrastructure, allowing data to be prepared for subsequent analysis tasks (e.g., initiation of start-up processes). The effectiveness of this approach is limited to the number of VATs used during an analytical task, as performance may degrade due to the complexity of managing multiple simultaneous preparation steps.

Order While the timeline of the data exchange process is used to organize the execution of data exchange, there are also different strategies governing the status of the data during the execution order. Di Lorenzo et al. [Di +09] mention two strategies that are based on the invocation of said VATs.

The *Pull Strategy*, on the one side, is based on frequent and repetitive requests that are initiated by the receiving VAT. For instance, in a data-flow model, an idle operator would periodically poll its predecessor(s) in the toolchain for new data to process. The polling frequency should thereby be set to be lower than the average update frequency of the data source itself to avoid unnecessary operations and to manage the load effectively.

The *Push Strategy*, on the other side, is initiated by the sending VAT. For example, in a data-flow model, an operator that has completed its computation would send its results to its successor(s) in the toolchain. If the successors are not explicitly known to the operator, it may perform a central broadcast to all operators, allowing each individual operator to decide whether to accept or decline the incoming data.

Availability Characteristics

So far, the data exchange has been considered as a process by which a receiving VAT obtains information from a sending VAT over the system's infrastructure. However, in order to carry out this process, it needs to be defined if data is even available for the exchange process by answering the questions:

- *Which data can be accessed?*
- *For which time period is the access granted?*

Restriction Data can be obtained in many different ways, whether through a file system or databases, by adding data values as URL tokens, or through inter-process communication. However, security restrictions can limit access to these sources as shown in Figure 4.10

In the case of *Full access* to data sources, information can be freely exchanged between tools across the system infrastructure. This is the simplest case, where seamless and efficient data transfer is allowed among VATs.

However, in some situations, *Restricted access* might be applied so that data exchange is throttled or otherwise constrained. One way of relaxing this restriction are tool-specific application programming interfaces (APIs) through which data can be requested and sent. Yet, not all visualization systems supply such an API.

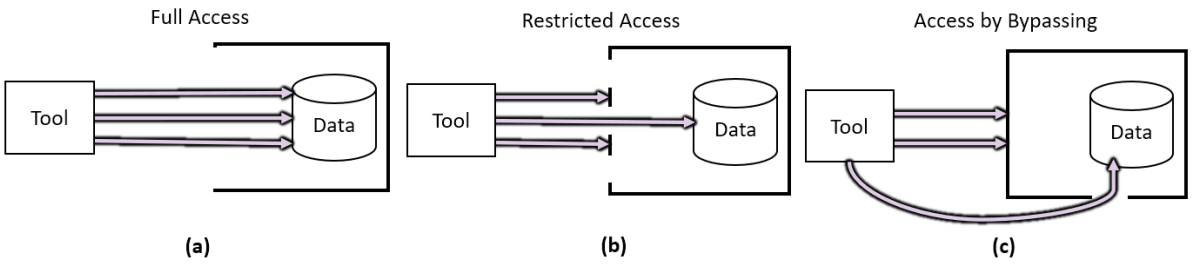

Figure 4.10: Visual representation for the distinction of access methods depending on the security restriction for (a) full access, (b) restricted access, and (c) access by bypassing to exchange data from one visual analytics tool (left) with the encapsulated structure of another one (right).

For otherwise proprietary applications, data retrieval is possible through *access by bypassing*. This applies to all situations where data is not meant to be exchanged so that minimally invasive workarounds like screen poking and screen scraping [Fer+11; HDK08] need to be applied. While screen poking is used to generate synthetic mouse and keyboard events for inputting data as if done manually, screen scraping is used in the opposite direction of extracting information from an application's UI. In the case of structured data being unavailable, techniques like Optical Character Recognition (OCR) or image processing [Mou+12] may be used to gain information. If such information is available – e.g. in the Document Object Model (DOM) of a website – an extractor or wrapper can be used to find and obtain the relevant information [GC09]. Another way of information retrieval is the parsing of rendered user interfaces [Fer+11] to extract content types from similar visual features in the synchronized views. Examples of toolkits for scraping data from different sources are the combinations of Firegoose [Bar+07], and the Gaggle Tool Creator [TBB10] or SideCache [DBR11] and SideKick [DYR10], which are used in the biomedical domain.

Even though most commercial applications limit the usage of their data, the overall trend in the scientific field seems to therefore lead towards shared information [AKD10]. Hence, this aspect might be of lesser concern in the broad scope of the exchange process.

Retention Beside access restrictions to data sources, there is also the problem of time-dependent availability as the retention of data might differ during the analysis task.

Persistent Access is granted to a VAT in order to retain data for an extended period, allowing the data to remain available for later use. This guarantees consistency during the analysis, as the obtained data is only modified by the VAT, which requests access until all needed operations are finished. However, this behavior will lead to conflicts if multiple VATs share access to the same data.

Transient Access is granted to a VAT for the duration of the exchange process itself. An example for this would be a unidirectional analysis process with a sequential use of multiple VATs one after another. In this scenario, there is no need to retain access to a data source, as the analysis tasks enforce a subsequent execution of VATs.

4.2.3 Combining Data Exchange Characteristics into a model for the Design and Evaluation of Visual Analytics Tool Ensembles

Combining all of the previously explained characteristics results in a comprehensive model that describes the properties of pairwise data exchange between individual tools at an abstract level.

The representation of this model shown at Figure 4.11 can be used to identify potential problems of data exchange without considering the underlying technology. This way, it can help to find criteria for the evaluation, comparison, and design of VATs. The feasibility of this approach is further described in one of the key submissions for this thesis [Non+20] to find the "space of possible" for three use cases. Through this, it provides ideas for further improvement in the data-flow model as a first step towards the necessary bridge between the visual analysis processes and technical coordination, which is further discussed in Section 6.1.

4.3 Orchestration of multiple pairwise-coupled Visual Analytics Tools

Up to this point, the structure and attachment of VATs are based on the assumption that tools are coupled pairwise, meaning there are two VATs connected to each other. However, domain expert workflows usually contain more than two and potentially unlimited numbers of VATs. This has to be represented through the coordination model accordingly. Hence, the following section describes the analysis process as chains of interconnected VATs – i.e., toolchains.

4.3.1 Definition of Toolchains and their underlying Structure of the Coordination Graph

A toolchain is the succession of specific tasks that can include multiple VATs that may be required to perform each step based on the domain expert's workflow. Each toolchain is, therefore, highly driven by the domain expert in pursuance of a particular analysis goal. The domain users workflow can include a high amount of subsequently or concurrently used sources such as data or tools from the application field.

In line with common view coordination models (e.g. Weaver [Wea05] and Collins [CC07]), we proposed to model these toolchains as a directed acyclic graph [Sch+19b]. This *coordination graph* describes the coupling between pairs of tools based on the linked data visualization model from Brunetti et al. [Bru+13]. In this graph, VATs constitute the nodes, and directed edges capture the properties of the previously defined usage flow, data flow, and control flow (see Subsection 4.1.1) between independent pairs of VATs as shown Figure 4.12. Each of the couplings can likewise be understood as sets of directed edges that, taken together, define a graph topology

	Aspects	Characteristics
Data Characteristics	*Content*	Data Set: a collection of values, items or facts
		Data Descriptors: data about a collection of values, items or facts
	Formats	Structured Data: follows a predefined data model
		Unstructured Data: follows no predefined data model
	Quantity	Full Data Exchange: exchanges the entire data in one big chunk
		Segmented Data Exchange: exchanges the data in multiple chunks
		Delta Data Exchange: exchanges only a modified delta partition of the data
Function Characteristics	*Operations*	Input / Output: exchanges data between tools to perform their basic operations on
		Modification: exchanges data to reflect changes
		Updates: exchanges data to incorporate new information
		Synchronization: exchanges state changes (selection, bookmarking, etc.) of data
Topology Characteristics	*Infrastructure*	Centralized Architecture: exchanges data through a single, unified platform
		Decentralized Architecture: exchanges data through a variety of different mechanisms
	Cardinality	One-to-One Relationship: data is initiated or exchanged between two tools
		One-to-Many Relationship: data from one tool is exchanged with multiple others
		Many-to-One Relationship: data from multiple tools are exchanged with a single tool
		Many-to-Many Relationship: data from multiple tools are exchanged with multiple other tools
	Directionality	Unidirectional Process: data is exchanged exclusively from one to another tool
		Bidirectional Process: data is exchanged both ways between two tools
Chronology Characteristics	*Timeline*	Synchronous Process: data is sent and received at the same time
		Asynchronous Process: data is sent and received at different times
	Order	Pull Strategy: data exchange is initiated by the receiving tool
		Push Strategy: the data exchange is initiated by the sending tool
Availability Characteristics	*Restriction*	Full Access: data can be freely exchanged between tools
		Restricted Access: data exchange between tools is throttled or otherwise constrained but still possible
		Access via Bypassing: data is not meant to be exchanged and needs to be exfiltrated/infiltrated using workarounds
	Retention	Persistent process: exchanged data remains available at any later point in time
		Transient process: exchanged data is only available for the duration of the exchange

Figure 4.11: Overview on the characterization of data exchange between independent visual analytics tools [Non+20], where each aspect is included in one of the five categories for data, function, topology, chronology, and availability characteristics.

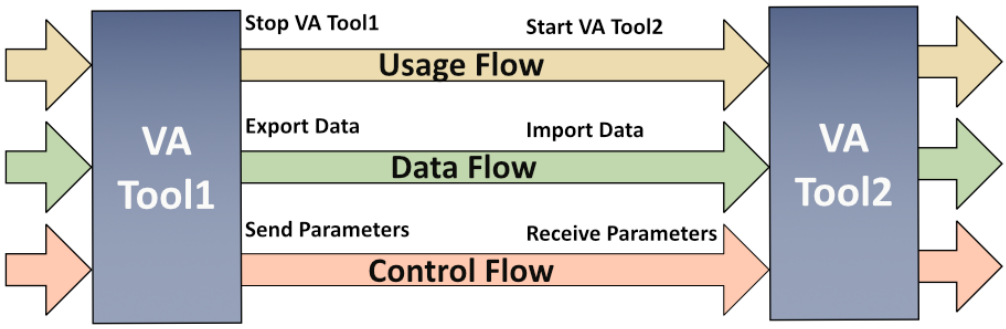

Figure 4.12: Visual representation for the concept for a coordination graph, where all flow-oriented visual analytics tools are pairwise coupled to represent the user's workflow as a sequence of analysis steps [Non+22].

over the set of VATs. The graph therefore models the interplay between tools and data through a sequence of links to define the temporal order of a particular workflow. Each tool of this workflow generates its own output that needs to be provided to the domain expert. They thus capture coordination in a bottom-up fashion by coupling two tools at a time and then combining these pairwise couplings into larger coordination mechanisms [Sch+19b; Sch+20]. Consequently, there is an inherent dependency between the flow-oriented coordination channels and the coordination graph. Hence, in order to create such a graph, the flow-oriented properties of the corresponding toolchain have to be described.

4.3.2 Dealing with Conceptual Separation through Layered Coordination

For further investigation towards the coordination of multiple VATs, the idea of layered toolchaining [Sch+20] was realized within an integrated visual analytics framework, which allows for a more flexible coupling of independent VATs compared to traditionally employed fixed pipelines. It binds otherwise autonomous VATs from loose multi-tool ensembles on aspects of their joint use to coordinate their coupling based on three separate concerns:

1. The *domain experts workflow* specifies the temporal order of tasks – i.e., the usage flow determined by the analysis scenario and its domain,
2. The necessary *data exchange* funnels data from one VAT into the next – i.e., the data flow determined by transforming data along the way and
3. The desired *synchronization* exchanges functional aspects between tools – i.e., the control flow determined by actions generating reactions in the other tools.

The approach for toolchaining is thereby *layered* in the sense that each of the individual levels of usage flow, data flow, and control flow can be used to affect a coordination among VATs either by themselves and with respect to the particular aspect of toolchaining or in combination with each other. An example of this would be two VATs that may only be coordinated with regard to their data flow. This means that users must still invoke one VAT after the other and manually parametrize them. However, the data exchange between them can be aided by a toolchain mechanism, ranging from injecting a tool for handling format conversion to fully automated, bidirectional live data exchange. Another example would be two VATs that are coordinated with regard to more than one level through their usage flow and their control flow. After closing one VAT, the next VAT in the usage flow would automatically be started together with an adequately parametrized view that reflects any manual fine-tuning done in previously used VAT, while the data output and input are carried out manually. This abstract representation of the coordination process still has to be handled by the role of a technical expert.

Describing the Usage flow

At the first step, the technical expert has to understand the domain expert's workflow in order to identify tool dependencies that should be automated via tool coordination. This step deals with the *situational level* in the nested visualization design model [Mun15] that aims to understand the domain situation. This includes selecting suitable tools for the given tasks, determining sequences in which they should be used, and guiding a domain expert through these sequences in the sense of automatically bringing tools up as they are needed.

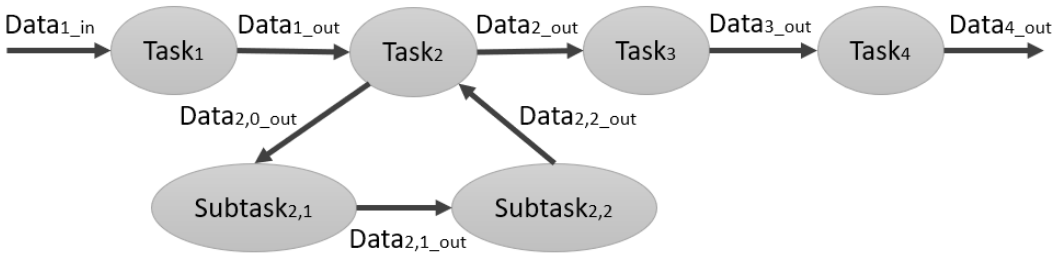

Figure 4.13: Visual representation of a task sequence consisting of four tasks with $Task_2$ containing two sub-tasks $Subtask_{2,1}$ and $Subtask_{2,2}$. The $Data_{i_in}$ and $Data_{i_out}$ of each i task are thereby considered to be from single or multiple data sources that are consumed or produced by single or multiple visual analytics tools.

As with every other temporal sequence, there is some ordering implied by a workflow. The first layer is therefore a sequence of tasks $Task_i$, that are executed subsequently or concurrently in order to achieve a goal or sub-goal of the visual analysis within the corresponding application domain based within the usage flow. Each task utilizes at least a single or potentially multiple VATs and optional data inputs necessary for carrying out

the tools – i.e., overview and detail exploration with a single framework or comparison of data with multiple visualization tools. This structure is recursive and therefore reusable in terms that resulting toolchains can potentially be employed as sub-tasks $Subtask_{i,j}$ with i being the task and j being the sub-task of a usage flow. This enables an abstraction of modular components to reduce the complexity of the toolchaining process while still obtaining the provenance over a hierarchical graph structure as shown in Figure 4.13. Furthermore this representation can unwrapped on the coordination graph as tasks can be simplified down to sub-tasks, where each node holds the corresponding VAT. The main goal of the analytical task is thereby represented by the highest hierarchical level of the graph. An example of such a usage flow sequence is shown in Figure 4.13. Through this, the technical expert is able to configure the toolchain, so that the right tools are brought up at the right time, as it is foreseen by the usage flow.

Describing the Data flow

Following up on the first step, the domain expert and the technical experts have to define how data is exported from one VAT to be imported by the next VAT in the sequence of the usage flow. This step deals with the *abstraction level* in the nested visualization design model [Mun15] that aims to understand the data transfer. The underlying data flow of a the system is thereby tightly related to the usage flow as VAT coordination automatically hands off data from one VAT to the next as the user proceeds with the analysis along the toolchain. Where the usage flow is determined by the chain of tools, the data flow is determined by the data funneled through this toolchain, starting from the 'raw' data input into the first tool of the toolchain and continuing all the way to the refined result output by the final tool in the chain.

This means that at the second layer, the means for exchanging data are therefore provided to make the analysis results from one VAT available in a form that can be read by the next tool in the chain (and possibly vice-versa, depending on how the tools are used in conjunction). The initial data input $Data_{in}$ is thereby processed and transported from one $Task_i$ to the next $Task_{i+1}$ as data output $Data_{i\_out}$. Coordination on this level automatically delivers the right data to the right tools at the right time, very much like data flow-oriented visualization frameworks do.

Describing the Control flow

At the third step, technical and domain experts must specify if interactively invoked actions appearing in one VAT shall be synchronized and therefore reflected in other VATs by exchanging their associated parameters and if so, how this synchronization should look like. This relates to the *encoding and interaction level* of the nested visualization design model [Mun15] aiming to find out how the general workflow is specifically carried out among tools beyond the mere passing of data. Where the data flow is determined by the inputs and outputs of VATs, the control flow is determined by which interactions are performed on them, which methods are selected, and which parameters are tuned.

For this, presets for parameters or functions that are used within the analysis toolchain

of an application domain are defined. This includes the identification of tasks and their translation into modular execution steps to provide the right parameters at the right time. At this level, VAT coordination automates the synchronization of interactive modifications across tools, which primarily involves the visualization or user interface of these tools as these are the common means by which the user interacts. The model captures the settings and interactive decisions made within tools, as expressed through their parameters, such as numerical filter criteria, color scales, or transformation functions, which can be tuned in one tool and re-used in another.

4.3.3 Dealing with Spacial Separation through the structured Use of User Interface Ensembles

When handling multiple VATs, the mere orchestration of their application windows becomes a management task by itself that requires attention, effort, and time which would be better spent on the visual analysis task. In order to reduce this overhead, we proposed the notion of *interface ensembles* [Sch+19b; Sch+20] (see Figure 4.14). These provide a structured and organized view on the otherwise often overwhelming network of hidden pairwise coordination mechanisms springing to life depending on the currently used tools and the actions of the domain expert.

Interface ensembles consist, first and foremost, of a centralized panel for global views and interaction elements that concern the toolchain as a whole. This centralized panel is called the *Control Interface* and serves at its core as a place in which to display information that cuts across tools and where to affect global changes and adjustments [Sch+19b]. In this function, the control interface can be utilized by all three levels of coordination:

- for the *Usage Flow*, one can use the control interface for example to display progress information and to invoke additional tools or choose between alternative analysis paths;
- for the *Data Flow*, one can use the control interface, for example, to archive snapshots of interesting intermediate results for later in-depth investigation along the lines of a bookmarking or multi-clipboard system;
- for the *Control Flow*, one can use the control interface, for example, to display global information like legends and to set global parameters like color scales or which data field to use for labeling data items.

In addition to the control interface, interface ensembles employ structured ways of displaying UIs so that they appear in a predictable manner that suits the current usage flow. Common ways of doing so are outlined in the following.

Individual Use: The Tabbed UI.

In this orchestration, the usage flow among VATs is a path. First a tool A is used, then a tool B, followed by a tool C, and so forth. This individual, subsequential use of VATs, as

predefined by the coordination orders that connect the tools in a temporal sense, results in the exclusive use of a single UI at each point in time as shown in Figure 4.14(a). What reads like an oversimplification at first is actually the most prevalent usage pattern in practice, as visual analysis is for the most part conducted as a linear series of very specific analysis steps. Starting from raw data to insight or from overview to detail, each carried out analysis usually ends up with a highly specialized analysis tool or view.

To the domain expert, these tool sequences can be presented in several ways. One way of displaying such sequential procedures is to use a tabbed interface, where each tool opens in a dedicated tab, with the tabs ordered according to the tool sequence. This allows the domain expert to revisit any earlier tool in the chain and modify settings, such as manually shifting a data item from one cluster to another. Given that all other parameters and settings along the toolchain remain unchanged, the adjustments can be automatically propagated through the appropriate channels to be processed by the appropriate rules and auto-update the current tool and its view. The tabbed interface can also be enhanced with a wizard-like guidance that leads the domain expert tab by tab along the path defined by the coordination orders.

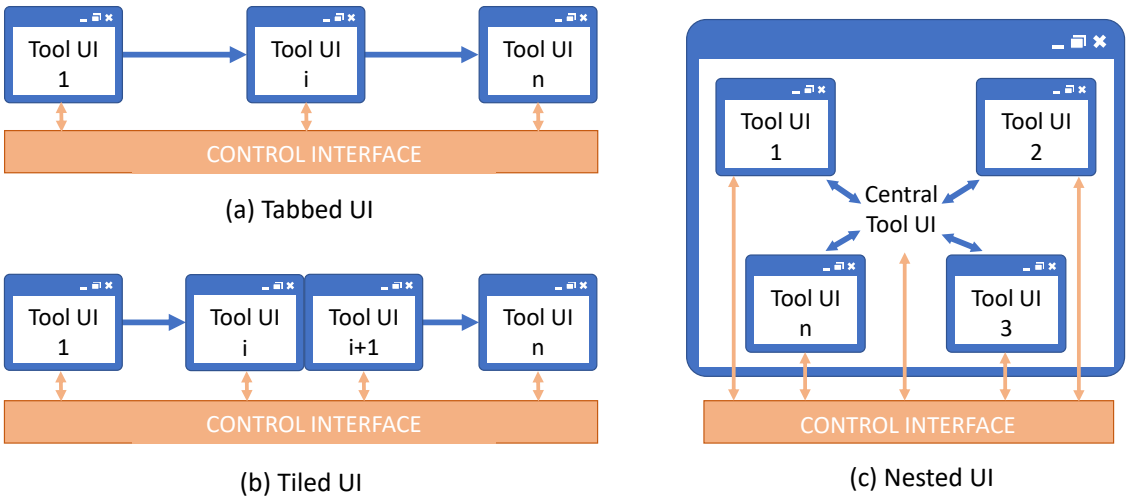

Figure 4.14: Common user interface layouts when dealing with multiple visual analytics tools [Sch+19b]

Combined Use: The Tiled Display.

Sometimes, it makes sense to use VATs not just one tool at a time but to have access to subsequent tools of the toolchain at once. This can be the case, for example, when a data selection from one tool will serve as an input to the next tool, and one needs to go back and forth between the two tools to try out and observe the effects of different selections. The topology is still a path topology, as shown in Figure 4.14(b), but with two UIs being displayed at once to facilitate such combined use.

To the domain expert, such setups are usually offered by tiling the display and showing the tools side by side or distributing them among multiple monitors. In this way, the

tools are present on the screen at the same time to work with them without having to switch – i.e., sending one to the background and bringing another one to the front, as it would be the case in the tabbed interface. Synchronization features, such as linking & brushing, and displaying visual links, are desirable to make the back-and-forth between the tools more fluent.

Flexible Use: The Nested UI.

If VATs are used more flexibly than a mere back-and-forth along a path of sequential tools, the resulting topology also becomes more involved. A powerful example of this case is the star-shaped topology that is shown in Figure 4.14(c), where all analysis steps start from a hub application or central VAT. Such topologies support more complex usage flows that meander between multiple tools until their combined use yields an analysis result. This is often the case in comparative analyses where multiple windows and tools are needed to process, show, and relate different data subsets or different analytical procedures to each other.

To the domain expert, the central tool is usually offered as an omnipresent overview of the data that is shown in a fashion similar to a background image. In this overview, users can select regions of interest into which to dive deeper by opening them up in other VATs. The opened tools are shown as nested or superimposed views right in the place where the selection was made. Making multiple selections opens multiple tools, effectively realizing the star-shaped topology. For this to work even with a dozen tools all scattered across the overview of the central VAT, the overview/background needs a map-like appearance [But+08] that serves well as a context for all the other UIs and makes their spatial relation meaningful.

In practice, VATs combine all of the mentioned UI ensembles, just as they may be needed during a particular stage of an analysis. As illustrated in Figure 4.14, it is not uncommon that the same tool may be instantiated multiple times for different parts of the data or for a tool's findings to initiate an entirely new sequential analysis workflow. In such cases, there is no principal way of how to best combine all UIs involved and it has to be negotiated with the domain expert. The more complex this ensemble gets, the more crucial it becomes to maintain a clear overview of the analysis to be able to parametrize and steer it.

4.3.4 Dealing with Temporal Separation through an interactive Toolchaining Configuration Interface

Based on the layered approach [Sch+20], we presented the idea of using a configuration interface to support domain experts and technical experts in their collaborative mission of configuring toolchains and controlling the analysis process [Non+21; Non+22]. It thereby follows the predefined user roles defined in Subsection 3.3.2, so that the domain expert can explain the tasks and the temporal order of the domain workflow while the technical expert connects VATs and structures the necessary transport and transformation of data throughout the process. In order to support domain experts in their task, this requires

1) access to all tools needed, 2) access to all data sources needed to run these tools, and 3) defined connections between tools and data sources in the ordering process.

Structure of the Configuration Interface

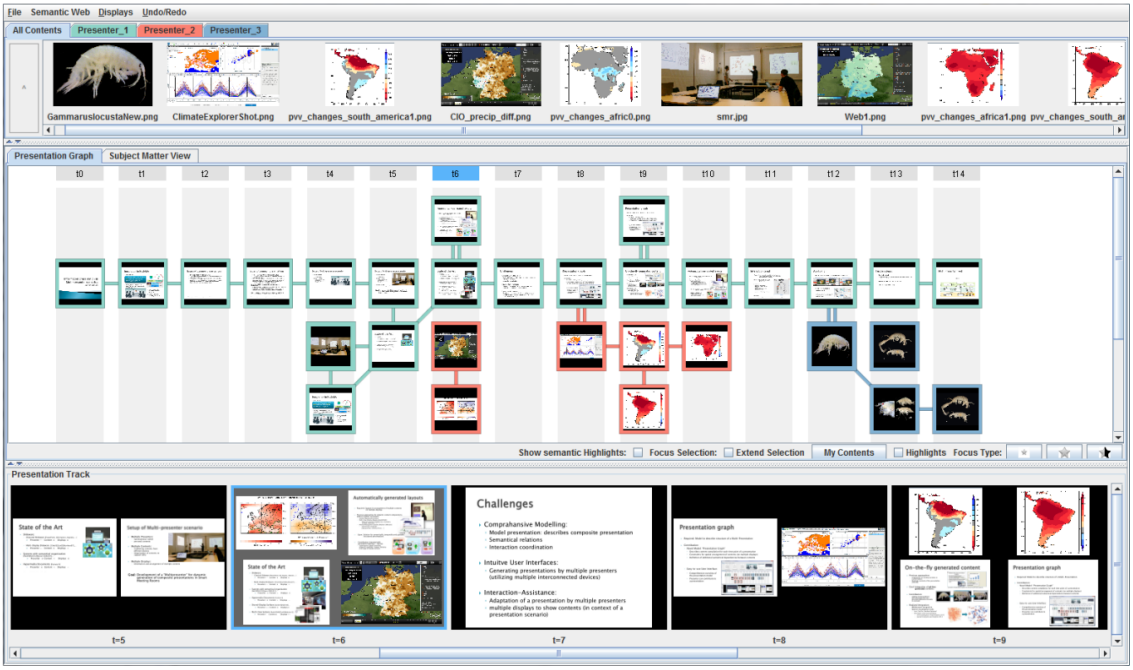

Figure 4.15: Graphical interface from Eichner et al.[EST19] that is used for the preparation and control in multi-display visual analysis sessions. Informational content is thereby imported into the content pool (top), organized in the logical session structure (middle), and displayed in the presentation preview (bottom).

To build a metaphorical bridge between the domain expert and the internal tool coupling processes, the configuration interface is similar to the interface of the presentation software, where a user can add and position a sequence of slides for presentations [EST19] (see Figure 4.15). The illustrated interface of this approach (see Figure 4.16) is divided into three containers that hold different types of information for the ordering process.

1. The *Tool Container* holds a collection of all executable tools that can be added into the graph container to either create new tasks or add VATs to existing tasks.

2. The *Source Container* holds a collection of data sources that can be included in the graph container to supply tools at a certain task with input or supplementary data items.

3. The *Graph Container* is the central part of the configuration interface, providing a space for the creation and modification of the coordination graph to directly

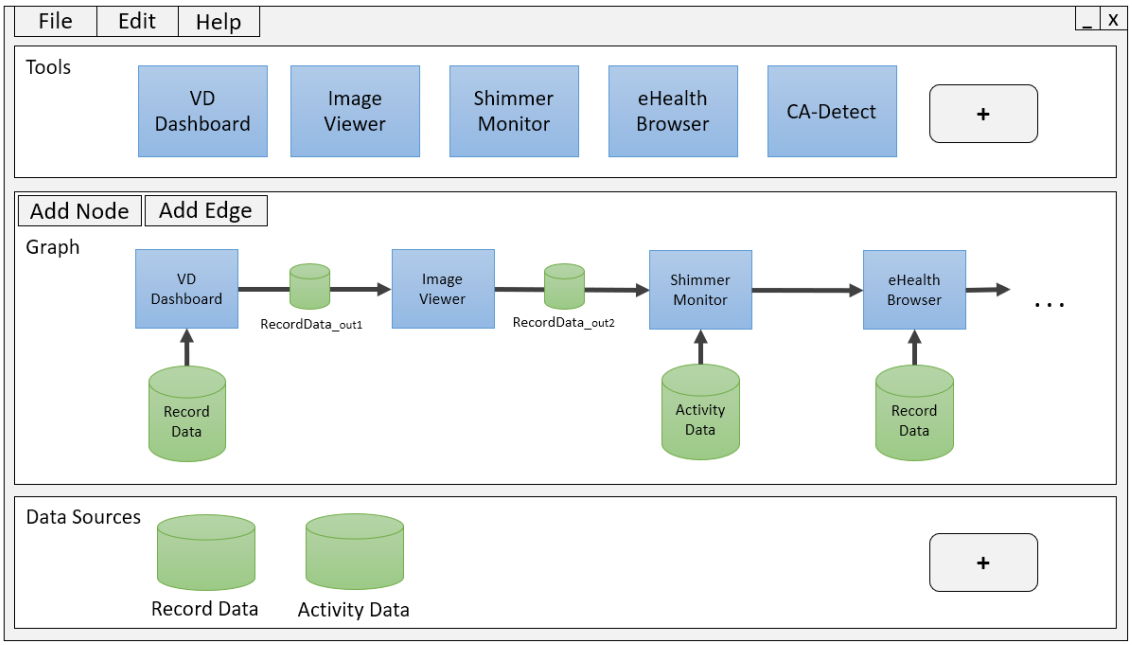

Figure 4.16: Conceptual draft of the configuration interface for a simplified scenario including the tools used in Health@Hand [Non+19]. The interface's panels include the tool container (top) holding all imported tools, the source container (bottom) holding all imported data sources, and the graph container (middle) holding a temporally structured combination of tools and data sources that are connected over edges.

manipulate tools and data sources. Once finished, the fully organized analytical process is represented through a temporal toolchain that can further be executed by the domain expert.

This approach is highly flexible with respect to the number of imported tools or data sources as well as to the size and complexity of created toolchains.

Structure of the Control Interface

While the configuration interface helps to build the metaphorical bridge, domain experts still require some sort of representation to traverse the underlying coordination graph that holds the structure of the defined flow-oriented behavior. A control interface can offer a global display of the analysis workflow and the user's progress in pursuing it [Str+12]. The control interface in this approach is considered to be a small overview window at the edge of the screen that includes both of the following functionalities.

1. *Process Overview* serves as the central hub, offering essential contextual information, such as the current stage of the analysis, the connectedness of tasks, and orientation towards the completion of the problem analysis.

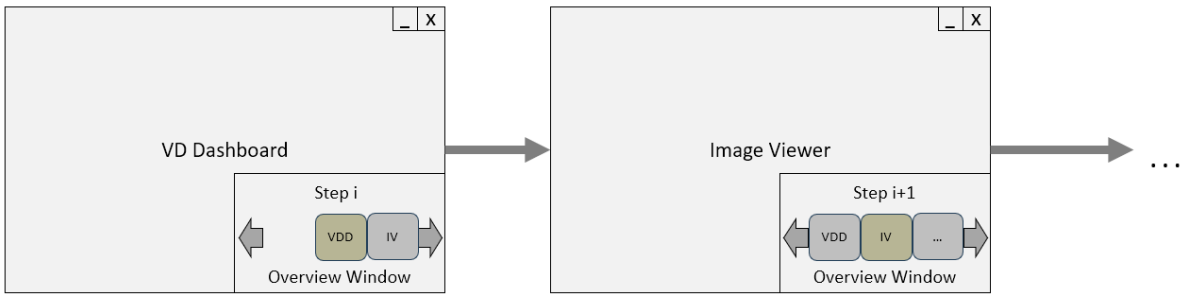

Figure 4.17: Conceptual draft of the control interface for a simplified scenario including the tools used in Health@Hand [Non+19]. The overview window (bottom-right) is thereby assumed to be in front of the current application of the toolchain. It includes a process overview in form for the current analysis step i with additional representations of previous, current, and upcoming tools as well as steering controls through arrows on the left and right.

2. *Steering Controls* empower users to navigate between different stages of the analysis, managing tasks according to their needs at hand or adjusting parameters or configurations dynamically in response to evolving requirements or insights.

Together, these components of the control interface synergize to streamline the analysis, enabling users to traverse complex workflows with confidence and precision. A representation of such a control interface is displayed in Figure 4.17. Comparing this view to the previously established configuration interface from Figure 4.16, it is clear that the control interface is much simpler in terms of controls. Yet, it still provides the necessary information and controls for the domain expert to engage with their analysis task and may be used without interference from the technical expert.

Approaches for Provenance and Interaction Documentation

There are a few systems that have implemented mechanisms to keep track of user interactions and store them in a history, allowing users to review, recall, and retrieve their actions [Hee+08; Dun+12]. Other efforts have focused on discussing and investigating how to support and capture insight provenance [Nor+11]. Harvest [Got+08], for example, tried to automatically capture a higher level of user intents and provided insight provenance based on user actions [GZ08; GW08]. Yet, only a few systems have explored the idea of multiple concurrent analyses or alternative hypotheses, where users can branch off into different scenarios, backtrack, copy elements from one analysis to another, or progress through various phases of the analysis, with the potential to revisit each phase while still keeping other results accessible. The proposed approach of this thesis, therefore, empowers domain experts by placing the provenance and documentation directly in their hands. It incorporates features (described in Chapter 5) that facilitate the tracking of user actions and decisions throughout the analysis process, enabling experts to easily document their thought processes and insights. This not only enhances the clarity of

the analysis but also supports better decision-making and collaboration, as experts can revisit and build upon their previous work.

4.4 Interim Conclusion about the Coordination Model

Overall, the layered model of this thesis engages with the previously defined challenges from Subsection 3.1.2 on multiple layers to provide:

- characterizations for usage flow, data flow, and control flow to provide solutions for the conceptual separation of the user mental model, the system's implementation model, and the represented model based on the task at hand,
- definition for data exchange and user interface ensembles to provide solutions for the spacial separation of multiple VAT, data sources, and user interface components that represent the analysis process and
- an editor for the configuration of lightweight toolchains to provide solutions for the temporal separation of task, data, and tool functionality.

By addressing these multifaceted aspects of VAT coordination, the overall concept provides a solid, comprehensive framework for the pairwise coupling of VATs in analytical toolchains while placing significant emphasis on the previously defined user roles (see Subsection 3.3.2). It is envisioned that this model sets the stage for future exploration and refinement of orchestrated analytical processes, ultimately leading towards more informed decision-making when working with multiple independent VATs in various application domains.

5 Building a unified Interface for the Orchestration of Visual Analytics Tools

This chapter contains details regarding the implementation of the proposed concept. These include primarily a flexible editor for the configuration of toolchains according to domain workflows and incorporate automated means for data exchange and transformations [HS19; Non+20; Non+21; Non+22]. This led to the ongoing design of a unified UI for effective display of all relevant information in a coordinated toolchain [Röh+23] with several extensions to automated view arrangement and transitioning as well as annotation support according to previous research in this field [EST19; Sch+21; Tom+21]. Together, the two prototypes represent advances in the coupling of VATs both on the data level and on the view level for a given analytical workflow. Accordingly, important design decisions for the flow-oriented data exchange and the structure of the prototypical user interface are explained.

5.1 Abstraction of Workflows for the Definition of Toolchains

Given the described model of this thesis, domain workflows are elaborated as a series of tasks. These tasks can be fulfilled by analyzing data sources with the help of available VATs. This results in a defined coordination graph that specifies the sequence of the usage flow. In certain application scenarios where established workflows already exist, the tasks can be used as an orientation for the temporal order of VATs. If there is no prior domain workflow defined, the order of VATs must be determined together with the domain expert based on the previously established role model (see Subsection 3.3.2). However, this requires a simple interface for modeling a temporal toolchain that can be understood by non-technical domain experts.

5.2 Overarching Structure of the Analytical Process Configurator

For the implementation of the described multifaceted model for the coordination of independent VAT toolchains from Chapter 4, a platform called the *Analytical Process*

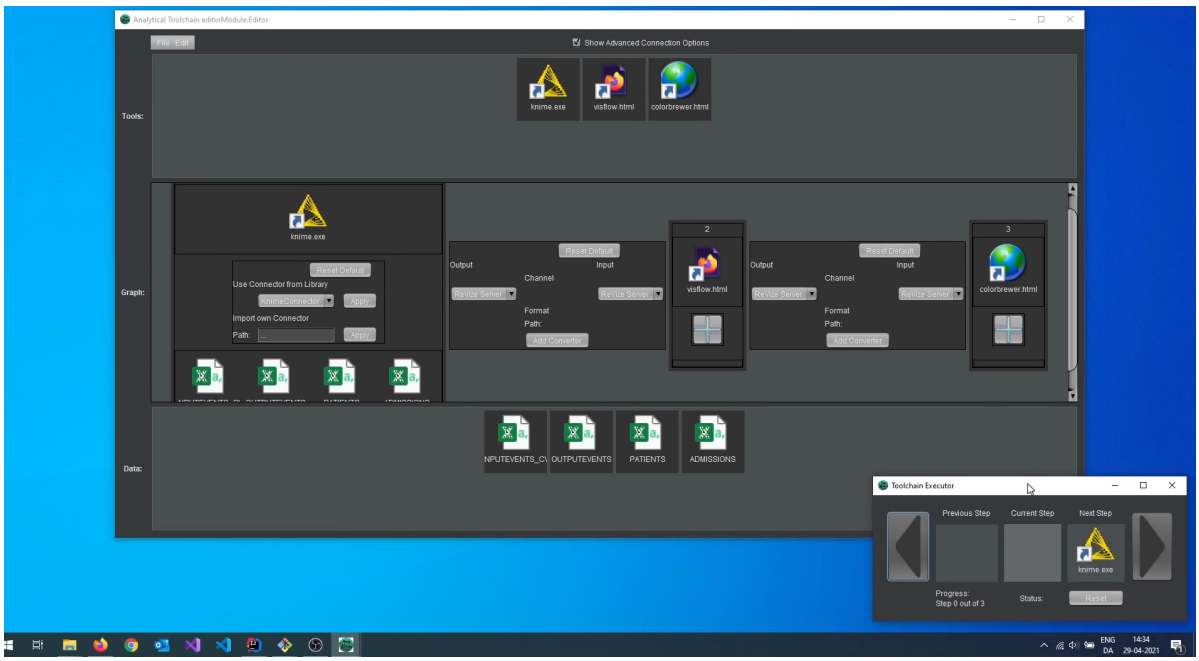

Figure 5.1: Screenshot of the AnyProc implementation showing the editor module (top left) and the excutor module (bottom right) that are used to define a toolchain for the visual analysis of the MIMIC-III data set [Joh+16].

Configurator (AnyProc) was created. This platform supports the described requirements for data coordination from Subsection 3.1.2 consisting of two major modules for the envisioned roles (see Subsection 3.3.2) to create an analytic toolchain based on the domain users workflow:

- an *editor* module for the technical expert to structure and configure the domain-specific toolchain based on the application and data exchange requirements,
- an *executor* module for the domain expert to carry out the visual analysis in a simplistic representation of the toolchain.

Figure 5.1 shows how these two modules can be used together in order to configure and execute the specified workflow, which is described later in Section 6.2. Each of these modules has different functionality and graphical interfaces depending on the purpose of the coordination process. Since each of these modules serves a different purpose and they are not equally important for the dataflow-oriented coordination, the structure differs significantly.

5.2.1 Providing an Editor Module for the Technical Expert

For the technical expert has to be supported in their task of connecting and coordinating data exchange between VATs. Following the established structure of the configuration

interface from Subsection 4.3.4, the implemented editor module consists of three different containers (see Figure 5.2).

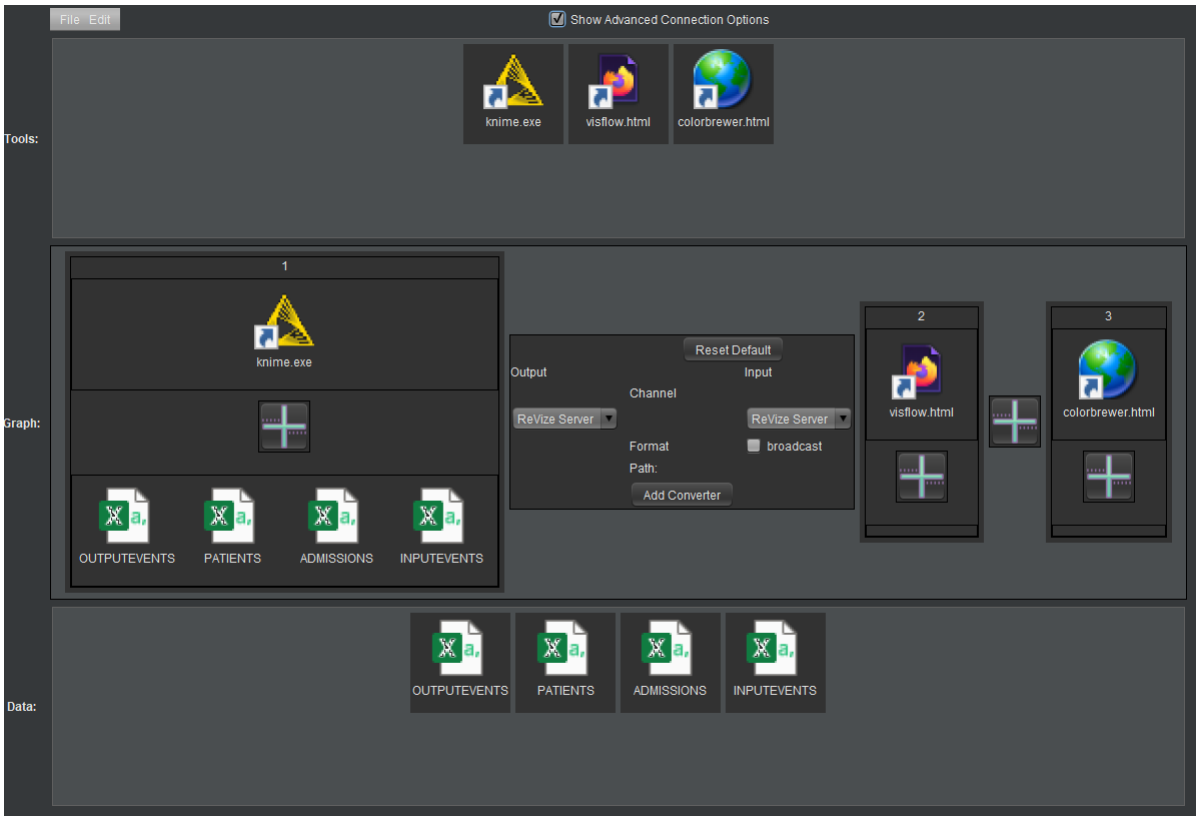

Figure 5.2: Screenshot of the AnyProc editor module with the VATs KNIME, VisFlow, and ColorBrewer (top) and the data sources patients, admissions, d_items, and output events (bottom) imported.

The top and bottom containers are used to import VATs and data sources accordingly, while the center is used for assembling the toolchain. All required tools and data sources can be easily imported into the editor by dragging them onto the two designated containers in the user interface. In fact, all import and export functions rely entirely on drag & drop interactions, enhancing the intuitive user experience. This design choice aims to help non-technical domain experts to quickly and easily adapt to the temporal ordering process of independent tools [Non+22; Sch+20].

After the import, the VATs and data sources can then be dropped in from the outlying containers onto the graph container to create new or add onto existing tasks of the usage flow. These tasks are represented as *process steps* in the graph container. This way, the technical expert can configure a sequence of task executions that aligns with the workflow of the domain expert. As part of this, each process step can encompass multiple VATs and data sources, which may be linked by drawing connections between the placed VAT or data source icons. This flexibility allows for the modeling of various cardinality characteristics [Non+20] facilitating both sequential and parallel executions involving single or multiple data sources.

By incorporating containers to hold imported data and VATs at the top and bottom, the technical expert can re-use these elements across various process steps within the toolchain. This not only saves time but also offers a clear overview in scenarios where the domain expert's workflow requires multiple instances of a VAT or the recurrent use of certain data sources.

The created toolchain can either be executed or saved as a JSON file by interacting with the main menu at the top of the user interface. While carrying out the execution, all VATs are opened according to their links with the associated data sources using the available communication channel. The connection of entry points for VATs with the corresponding data channel is further explained in Section 5.3 Overall, the editor reduces the cost for the technical expert to link various VATs and data sources.

Other important criteria in the development of a prototype implementation are the adaptability and expandability (see Section 3.3) of the application to guarantee independence as far as possible. While the basic structure of AnyProc allows the coordination of independent VATs, the expandability had to be enabled through different adaptations of the user interface.

Given that certain VATs may not be automatically compatible with the required data sources, it might be necessary for the technical expert to make further adjustments. To accommodate for this, a variety of data characteristics both during transport and import have been introduced through the option to include connector programs on each link established between VATs and data sources (see [+]-Button in Figure 5.2). Technical experts can leverage this option to incorporate custom utility tools that support the import of data into suitable formats for a specific VAT.

5.2.2 Providing an Executor Module for the Domain Expert

After the connection between VATs has been established through the coordination graph, the execution of the individual process steps has to be adequately supported. Here, it is important to keep the domain expert informed about the current tasks at hand while also offering navigational assistance in terms of what has been done before and what comes next.

Following the established structure of the control interface from Section 4.3.4, the executor module consists mainly of interaction parts for guidance and navigation in order to traverse the created toolchain. Utilizing the left and right arrow buttons shown in Figure 5.3, the domain expert can either initiate the next or revisit the previous VATs with the linked data sources according to the settings in the editor module. The established data flow will then be carried out, whether this means to simply open the same data file in another tool or to contact a server for a more elaborate data exchange. In this way, AnyProc is able to generate assurance of operations, as the domain expert can track the current progress along the toolchain and effectively plan further ahead or revisit earlier steps. Since the focus for AnyProc was on establishing connections for the coordination, the executor module has been presented in a very simplified way and was later integrated into the functionalities of a unified UI [Röh+23].

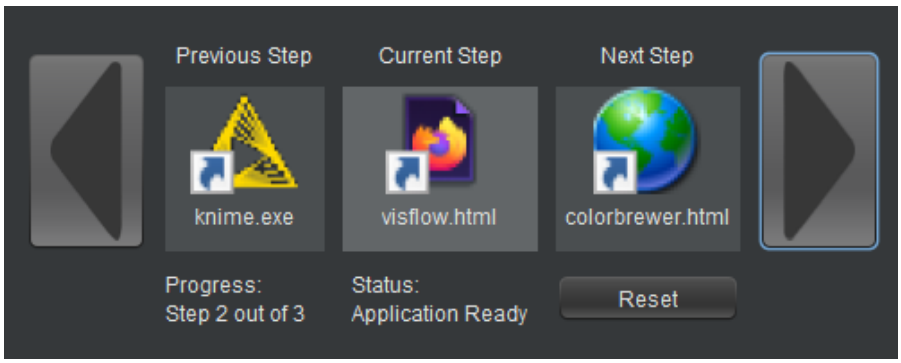

Figure 5.3: Screenshot of the AnyProc executor module showing the VATs KNIME, VisFlow, and ColorBrewer (center). The currently opened tool (VisFlow) is slightly highlighted by a lighter color. The subsequent or previous steps of the toolchain can be opened via the control elements (arrows left and right).

5.3 Approaches to flow-oriented Data Exchange

The basic structure of AnyProc allows for modeling diverse toolchains that can be executed with the default system settings. Nevertheless, as mentioned in Subsection 3.1.2, this is often not enough. To address this, an optional setting was introduced that enables the technical expert to directly control the data transport between independent VATs. Positioned between each process step are so-called `ConnectionNodes` for independent adjustments. For instance, the technical expert can modify the communication channels used for data exchange, switching from web sockets to the file system or clipboard (see Dropdown Menu in Figure 5.4).

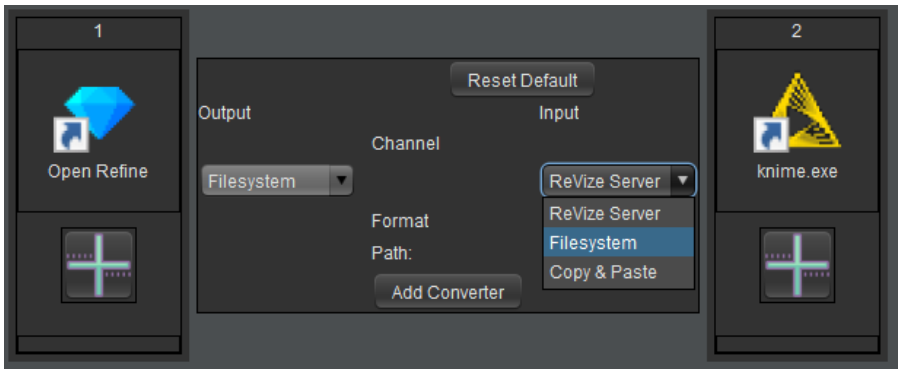

Figure 5.4: Screenshot of the `ConnectionNode` within the editor module showing the specification for the connection between OpenRefine and KNIME. Data channels can be adjusted through drop-down menus, and data converters can be added using the [Add Converter]-Button to resolve any mismatches.

Moreover, the technical expert can add their own converter configurations to be used as utility tools between the process steps to perform data transformations before the information is transmitted along the chosen communication channel (see [Add Converter]-

Button in Figure 5.4).

5.3.1 Finding Means for Data Exchange between Independent Visual Analytics Tools

As previously discussed (see Subsection 4.2.2), data exchange between multiple VATs is not as straightforward as it may seem due to the different types of data being generated. While some tools produce numerical outputs that are not necessarily visualized, others produce visual outputs as it is usually grounded in the nature of Visual Analysis. While numerical output can at any time be transformed into visual output, the other way around is hardly possible if another tool requires its input in view form. This leads to situations, where later on in the process, changes need to be made to analysis steps carried out on the numerical data. For example, when should the data have first been normalized, or does some aggregation parameter need to be adjusted? If this happens during a visual-interactive step, the whole toolchain needs to be rolled back to said point to re-enact the desired change and then be carried out again from this point onward. This challenge is usually countered with a linking between any shown graphical object and its underlying data item, so that any interaction and manipulation in the view space can be directly reflected in the underlying data space. An example of this would be employing centralized architectures using the Model-View-Controller(MVC) pattern so that any interaction and manipulation in the view space can be directly reflected in the underlying data space. However, this requires deep integration with the coordinated tools that go far beyond the mere input and output of files and hence conflicts with the initial requirements (see Chapter 3).

To instead realize a lightweight data exchange that follows the idea of pairwise connections between tools through the passing of files, this approach relied on the visualization grammar Vega-Lite [Sat+17] to describe the entire visualization pipeline consisting of data, data transformations, visual mapping, and view transformations all at once in one file. Consequently, VATs that can read and write such grammars can adjust any aspect of a visualization, from the underlying dataset to color scales and axis labels in any order. By providing the open-source JavaScript library *ReVize* that equips web-based tools with Vega-Lite import/export functionality [HS19], an abstraction of Vega-Lite encoded visualizations is made available to make the contained view hierarchy and data transformation graph accessible. In joint consultation with the University of Aarhus, we chose Vega-Lite, in particular, as a standard interface for the data exchange for multiple reasons. First, all static parameters and operators of the visualization pipeline are accessible to any tool, meaning the same channel can be used at any step in the toolchain. Second, grammar serves as a standard data exchange format, meaning that any tool, such as "speaking," that language can be used (in any order) in the toolchain. A third benefit is that while using relatively simple grammar to construct visualizations, Vega-Lite has been adapted to many use cases and is used in many tools.

Using Vega-Lite's view composition structure and decomposing it into independent layers for individual modification, ReVize's import module is able to resolve structural

dependencies (e.g., inline datasets or inherited visual encodings) that usually prevent the re-use of sub-views in Vega-Lite specifications by inferring default values from parent and child nodes. Thus, views on any level of a view hierarchy can be exported as independent Vega-Lite specifications. A VAT using ReVize can thus import a Vega-Lite formatted visualization description, further process its contained data, or interactively adjust the view on the data it describes and export it as a Vega-Lite specification again.

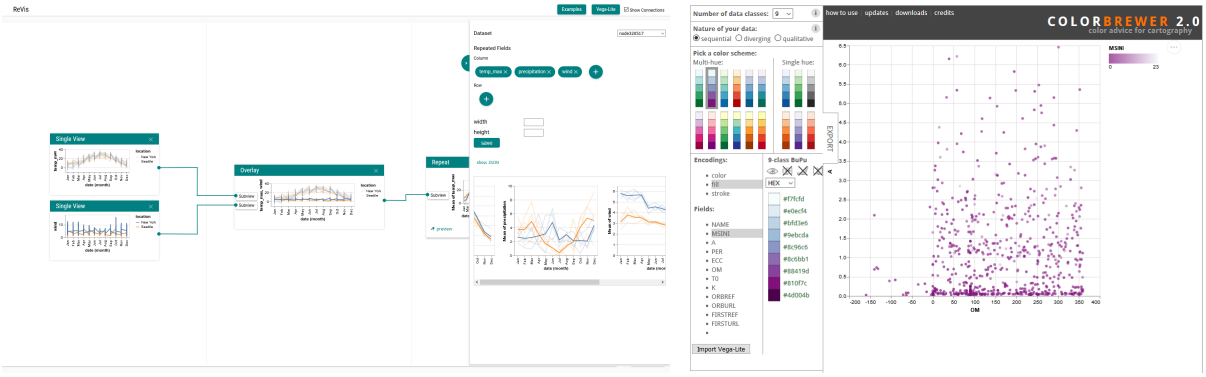

Figure 5.5: Screenshots of ReVis (left) and the Vega-Lite enabled ColorBrewer [HB03] (right)

ReVize can, on the one hand, be used to build dedicated VATs tailored to processing or adjusting individual aspects captured by the Vega-Lite format. This was utilized for building a data editor called *ReData* to insert/remove data transformations, and a visualization editor called *ReVis* to create/adapt the visual mapping and thus the display of the data in Figure 5.5(left). On the other hand, ReVize can also be used to add Vega-Lite input/output functionality to pre-existing tools, thus enabling us to use them as part of a Vega-Lite-based toolchain. An example of this would be the application of ReVize to augment the widely recognized ColorBrewer web application [HB03] with Vega-Lite import and export functionality so that it can be used to modify color scales in Vega-Lite formatted visualization data (see Figure 5.5). Together with all the tools that already feature Vega-Lite input and/or output, these VATs form a versatile toolset from which analysts select the most suitable option for their subsequent analysis steps. Besides the expressiveness of Vega-Lite for numerical and image data alike, there are more benefits from data exchange that emerged when working with this use case. First, by employing a generic data exchange format that is not tailored to a particular application domain, it is feasible to integrate domain-specific tools like ColorBrewer into the toolchain, despite its original intent for cartographic applications. Second, by capturing all aspects of an intermediate result from the toolchain in a single file, these files can simply be shared back and forth with collaborators from other departments to clarify questions or make adjustments to the data part of the file with the specialized domain tools. Finally, a simple form of cross-tool undo/redo functionality comes for free with this type of coupling, as snapshots of files can easily be archived every time they are passed into the next tool.

Feature	Supported by AnyProc?
Datasets	Yes (via all channels)
Filters	Yes (via Vega-Lite)
Data transformations	Yes (via Vega-Lite)
Models	No
Visual encodings	Yes (via Vega-Lite)
Selections/brushes	Yes (via Vega-Lite)

Table 5.1: Table with parts of the visualization that can be shared between tools via data exchange channels in AnyProc.

5.3.2 Coordinating the Data Exchange between Multiple Visual Analytics Tools

AnyProc's connection channels further describe the necessary data exchange between tools via web sockets, the file system, and the clipboard [Non+22]. The primary channel for data exchange in AnyProc is ReVize *websockets* [HS19; Non+21], which allows tools to automatically send and receive data to and from each other via a TCP connection. To achieve this, all visualization tools communicate with a central server that distributes data to other tools. The main benefit of this channel is the complete automation it provides. The domain expert simply launches VATs using the executor panel, and AnyProc automatically exchanges data between them, allowing the domain experts to continue their work without thinking about importing or exporting data. Moreover, web sockets provide flexibility in the way in which tools can be used. For example, there can be multiple tools exchanging data at the same time.

With ReVize, visualization tools use web sockets to import and export JSON objects that contain declarative visualization grammar descriptions of the visualization pipeline. A detailed overview of the capability of the websocket interface can be found in Table 5.1.

To make a VAT "ready" for exchanging data over web sockets, it needs to fulfill two conditions:

1. It needs to be able to read and write TCP web sockets, for instance, using the `socket.io`[1] library, which is available for many programming languages.

2. The VAT needs to be able to import and export Vega-Lite specifications.

For browser-based VATs, both can be facilitated by the ReVize library [HS19].

Whenever a tool does not support the websocket channel, AnyProc provides fallback options to nevertheless automate parts of the data exchange. The first fallback option is the *file system*, i.e., two tools exchange data by exporting and importing it to a file accessible to both. The general procedure for this channel is that the domain experts store the data in a file using their current tool, specify this file as input to the next tool using the AnyProc editor, and then open the next tool using the executor panel.

[1] `https://socket.io/`

This means that, in any case, domain experts have to export their data manually via the interface of their respective tools. For some tools, however, AnyProc can automate the data import, for example, by passing the file URI as a parameter to the next tool. For tools that do not support passing a file URI as a parameter, technical experts can provide connectors that extend the functionality of the tool to automate the import. If no connector can be provided, for example, whenever the source code is not available or implementing one, data import can still be performed manually.

Since not all tools may have access to local storage, using the file system may not always be a feasible option, either. Thus, another fallback option to web sockets and the file system is to rely on the *clipboard* to exchange data between tools via copy-and-paste. For instance, when integrating a visualization tool that accepts the Vega-Lite grammar but does not support the ReVize web socket channel described earlier, technical experts can manually copy the JSON description from one tool and paste it into the next. Certain steps in this process can be automated, such as copying data to the clipboard from the previous VAT or automatically opening the input field in the target VAT through scripting. Nevertheless, despite these automation approaches, clipboard-based data exchange remains predominantly a manual process.

5.4 Approaches for View Layout Configuration

The layout of view proves to be a challenging task as it not only includes the decision for the right UI ensembles (see Subsection 4.3.3) but also additional choices such as the proper aspect ratio or visual highlighting to emphasize the importance of a VAT in current step. This might even be in a semantic relationship to the underlying metadata within tools or data sources. It is therefore important to support the combination of VATs and the access to data and functions of all tools to be used within a given analytical workflow.

Prototyping View Layout Management in a Comprehensive Application

While working on technique-centered strategies for the management of data and view (see Section 6.1), we developed a toolchain module (see Figure 5.6) for the *Health@Hand* interface as a modified version of the underlying Plant@Hand3D system [ASU13] The visual interface is thereby based on the nested UI model, providing a general overview of the 3D scene with relevant short information about each element. The domain expert can interact with this UI by selecting a single short information panel to open up more detailed information. The detail information panel can contain text, images, or entire tools to further examine selected instances of a data source. The individual tools can be used independently to handle the data from typical healthcare management systems.

The *Health@Hand* system uses different sorts of parameters in the data manager to control, for example, 1) which information is shown in the 3D model, 2) how colors are adjusted to avoid visual clutter, 3) how screen space is allocated and organized for the parallel usage of multiple tools, and 4) which tools are linked in order to identify

Figure 5.6: Toolchain view of *Health@Hand* with the toolchain window (bottom right), the history window (bottom left), and the currently used tools (center).

correlations in the visual representation [Non+19]. The data manager can be used to switch between different data sources. However, there is also a possibility to select or deselect different parameters as every element of a data source is passed as a parameter to create the short information panels. The domain expert can thereby interact with a side menu in order to avoid visual clutter. The toolchain module itself is a part of the data manager and consists of three different components:

The **Toolchain Window** provides information about the current progress of the domain expert by showing an overview of the analytical toolchain graph. The graph can be explored by interacting with the buttons below in order to switch back and forth in the analytical process. The domain expert can also find helpful information about upcoming tools by hovering over the related steps.

The **History Window** holds a record of all performed operations, whether it is inside of a tool or by interacting with the toolchain window. These operations can then be reverted or repeated on demand.

The **Workspace Area** allows domain experts to interact with the corresponding tools of each toolchain step. It adapts the size of each tool according to the screen space and organizes them next to each other to avoid overlap. The referred data for the scaling and positioning of each tool is thereby passed as a parameter at the initiation of the next toolchain step.

5.4.1 Enhancing the Executor Module for Visual Overview

In a more recent approach [Röh+23], we extended the limited functionality of the existing AnyProc executor module (see Subsection 5.2.2) to support toolchaining for retinal VATs [Röh+23]. Together with experts in the field of ophthalmology, we have worked on an evaluation process (see Section 6.3), which provided requirements for the user interface design that can not be met by the simple executor prototype. From the user's point of view, it is important to get an idea of the work at hand before executing a particular workflow. This involves not only the individual workflow steps that must be performed but also the data to be analyzed and the extent to which tool coordination is supported.

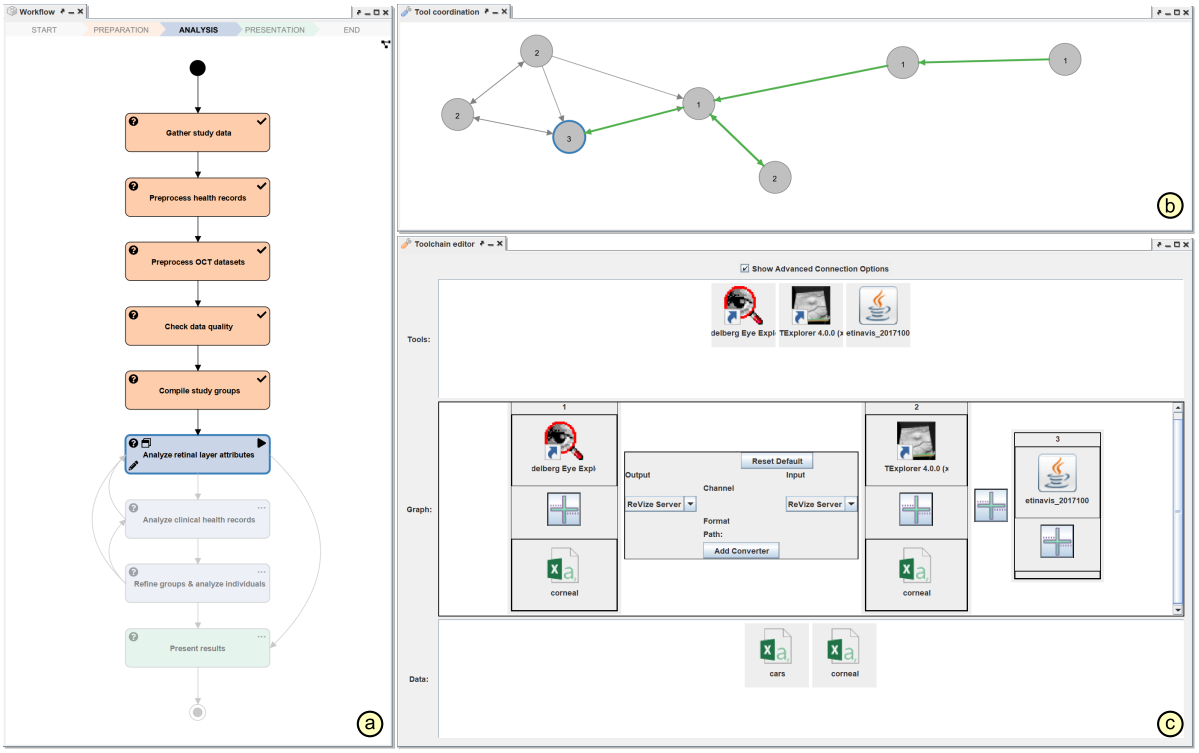

Figure 5.7: Visualization of workflow, coordination graph, and toolchain editor. The workflow steps are displayed as colored rectangles (a). The coordination graph is represented as a network visualization, where the circles depict the tools and the lines encode the links between them (b). The toolchain editor (c) consists of three panels showing the available tools (top), the data sources (bottom), and the data connections between tools (middle).

Therefore, a new overview UI was created consisting of three different visual components (see Figure 5.7) that enable users to view individual links between VATs in detail and make adjustments as needed.

The *workflow view* displays the workflow steps as a flowchart (see Figure 5.7a). The steps are shown as rectangles colored according to their workflow stage of preparation, analysis, and summarization. The arrows between the rectangles encode the possible

back and forth between the steps. Information about the current progress within the workflow is provided by highlighting the steps and links that have already been performed.

The *coordination graph view* shows the toolsets used in the workflow as a network visualization (see Figure 5.7b). The toolsets are coded as circles, and the lines indicate their data exchange and activation capabilities. Hovering over the circles displays additional information, e.g., which tools are included in the respective set and in which steps they are used.

The *toolchain editor view* is based on the previously described AnyProc editor module to establish a connection between the workflow and the coordination graph (see Figure 5.7c). Users can either create a coordination graph for a given workflow and set of tools or specify the data sources to be used as input for a planned execution of an existing toolchain.

Together, these three views provide an overview of the work, the tools, and the data involved through a unified UI and allow VAT coordination to be set up as required.

5.4.2 Adding Traceability of Analysis Results

Aside from showing the user their results in the current, previous, and upcoming tasks, another requirement from the application-centered evaluation Section 6.3 was to document the work and progress. As illustrated in Figure 5.8, the new interface supports this by showing current steps in detail along with additional information about the coordination.

To focus on the current steps during workflow execution, the workflow view transitions from an overview to a detail view (see Figure 5.8a). This expanded view emphasizes the active step, providing a description of the task and showing how the relevant tools are arranged on the screen. The previous step is displayed above the active step and the next possible steps are displayed below it. Clicking on these steps allows to go back and forth within the workflow. Once a workflow step has been selected, the associated tools are automatically activated, and their content is arranged on the screen according to predefined layouts. Data transfer between the previous and current tools is handled automatically, as specified in the coordination graph. This streamlines the process, eliminating the need of manually searching for the right tools, activating them, transferring the work data, and customizing where the content is displayed on screen. To see the actual path taken so far through the workflow and coordination graph, the user can switch back to the overview visualizations (see Figure 5.7) at any time. This highlights the links of the active step and tools in the overviews, making adjustments in the coordination graph possible on the fly.

During the analysis, users can set the status of the active workflow step (e.g., pending, done, paused, canceled) to indicate the state of the work. In addition, notes can be added to the steps describing how the work was performed and whether it was necessary to deviate from the original work description (see Figure 5.8b). Furthermore, intermediate results can be captured via screenshots of the tool content currently visible on the screen

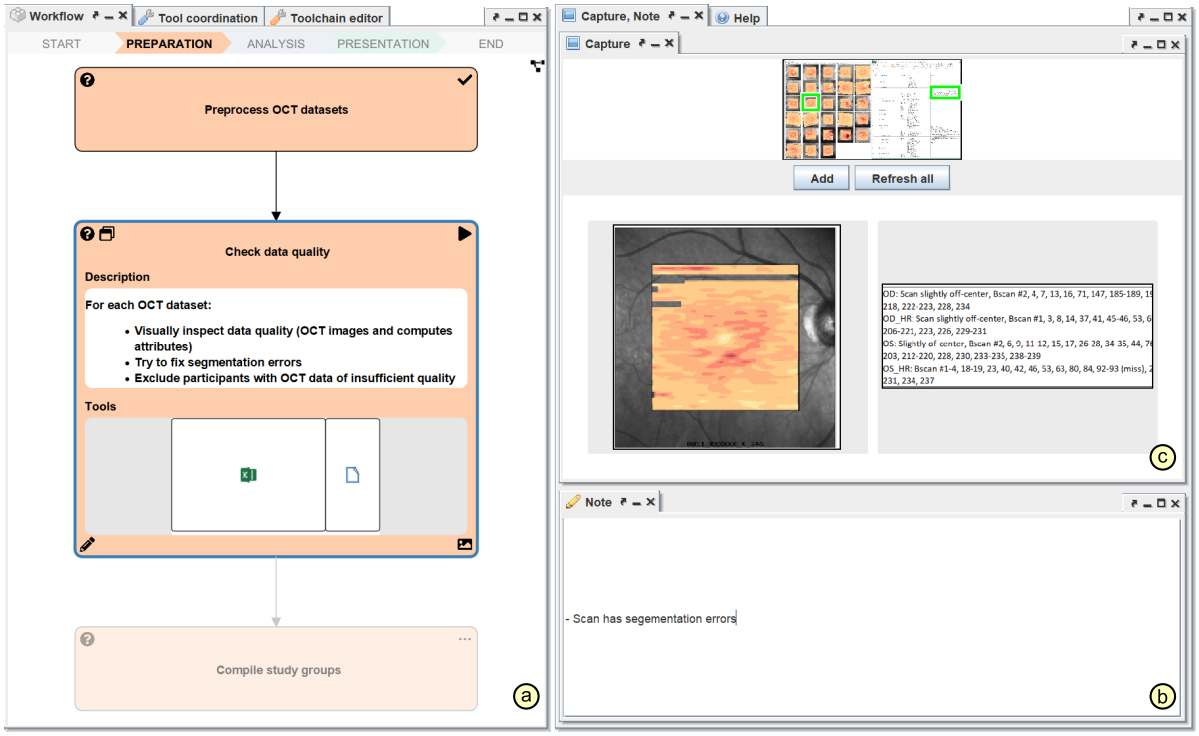

Figure 5.8: Visualization of individual workflow steps and additional information. On the left side, the current step is enlarged to show additional information, and the previous and next possible steps are displayed above and below (a). On the right side, a full description and user notes about the current step are depicted (b). In addition, intermediate analysis results are displayed as a list of screen captures (c).

(see Figure 5.8c). Multiple screenshots can be added by selecting any areas of the screen, with a reference image indicating which areas have already been captured. The content of the selected screen areas can be updated or removed again if relevant changes occur during work. If a workflow step is visited multiple times – e.g., by navigating back and forth in the workflow – a new set of notes and captures is created for each activation. That way, a detailed history of all work performed is generated. Especially when different paths are taken through the workflow, this helps to recapitulate which steps were taken to arrive at the results achieved in each case.

5.4.3 Storing and Displaying Information

During and after the execution of the workflow, it typically becomes necessary to summarize the work performed. Yet, navigating between workflow steps and multiple tools can make it challenging to recapitulate the most significant insights from various intermediate results. In fact, the end results do not have to come from a single workflow step and a single set of tools. It is rather often a mixture of intermediate results from multiple steps and tools that need to be reconciled and compared to provide an overall

picture of the work done and insights gained. To address this, the advanced interface provides an additional *summary view* for compiling results across workflow steps and tools as illustrated in Figure 5.9.

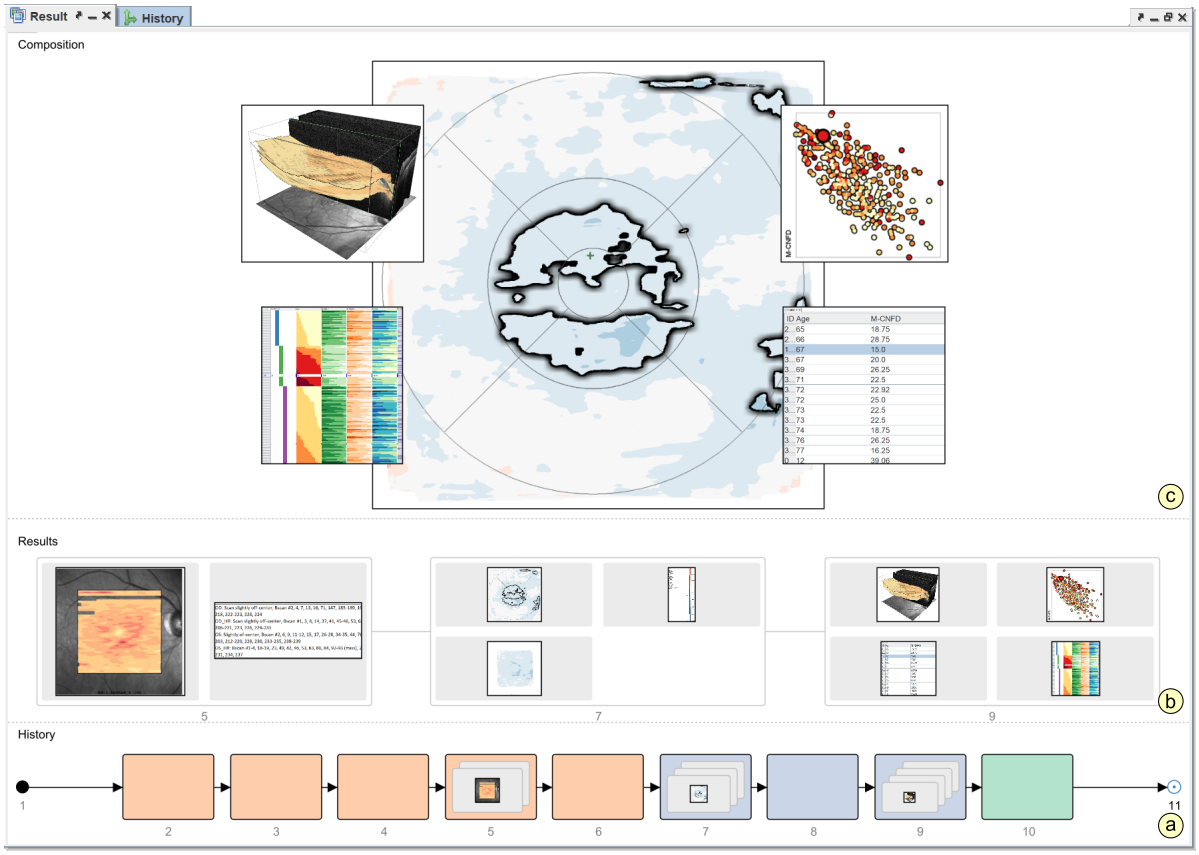

Figure 5.9: Visualization of workflow history and result composition. At the bottom, a history of all performed workflow steps is displayed (a). In the middle, all captured intermediate results are listed (b). At the top, selected intermediate results are summarized in an overview that illustrates the main findings of the data analysis (c).

The summary view is divided into three sections. In the bottom section, a history of all performed workflow steps is displayed in the order of their activation (see Figure 5.9a). Each step is represented by a colored rectangle with small icons indicating whether intermediate results have been captured. In the middle section, the intermediate results are listed and grouped by the respective steps (see Figure 5.9b). In the top section, the intermediate results can be interactively combined into a single image via drag & drop (see Figure 5.9c). The finished image can then be exported for further use. This enables a comprehensive comparison of critical findings from the analysis. Without the unified UI, this would hardly be directly possible if only the work steps and the associated tools were considered individually.

Overall, the different views of the unified UI establish a visual connection between

workflow, tools, data, and displayed content. The presented design approach thus assists users in creating a coordination graph that fits a particular workflow and provides valuable information for executing the workflow and understanding the results. To assess the utility, this approach was tested with a use case in ophthalmology described in Section 6.3.

5.5 Technical Note

AnyProc was developed and implemented as a native Java project with swing user interface components. This is mainly due to the pre-planned collaborative development of a unified user interface together with existing frameworks from Dr. Martin Röhlig. In retrospect, developing a web-based application (e.g., via React or Next.JS) in combination with electron (for the native file system support) might have been more purposeful for the data exchange integration, as this would have enabled us to more easily include existing functionalities from web-based services such as ReVize. Overall, the employed implementations for the presented model of this thesis cover all requirements from the previous analysis (see Section 3.3). Thus, the conceptual idea of a multifaceted model for the orchestration of VATs based on the separation of concerns has successfully been transitioned into a prototypical implementation.

The AnyProc platform and the additional mentioned ReVize tools are available under a permissive open-source license and linked as a footnote on this page. [2] [3]

[2] `https://github.com/nonnemann/anyproc_public`
[3] `https://vis-au.github.io/toolchaining`

6 Three-staged Evaluation on the Applicability of Coordination Mechanisms

This chapter provides a stage-wise evaluation of the previously established research questions, based on both the theoretical concepts of this work and the practical implementation, to prove or falsify the stated hypothesis. To this cause, the presented coordination methods are assessed against three use cases, each with a specific VAT set, a concrete domain workflow, and clear analysis objectives. The conducted evaluation of this thesis is thereby performed from three different angles:

1. a formative [technique-centered evaluation] to test developed coordination mechanisms and view layouts with a user group consisting of computer scientists, software developers, and user interface designers in order to gather feedback about the established methods,

2. a [workflow-centered evaluation] to combine existing individual VATs in coordinated toolchains with a user group consisting of VA experts in order to provide unified access to different tools and data sources,

3. a [application-centered evaluation] to present a functional prototype and compare it to manual toolchaining with a user group consisting of domain experts in order to solve analysis tasks and find hurdles to the practical adoption of the developed concepts that may need to be addressed in the future.

Overall, the technique-centered evaluation is performed from the engineering perspective to help improve the design and methods developed, while the workflow-centered and application-centered perspectives are more summative in nature with the goal of leading towards new insights and prompting new ideas for further improvement. In fact, these phases of the evaluation are meant to specify interesting topics for future research as the user feedback should draw on previous evaluations, but most importantly, on a wide network of staff with different backgrounds.

6.1 Technique-centered Evaluation

At the stage of technique-centered evaluation, different approaches concerning data and view management have been tested in isolation. One prominent result of this is the

Health@Hand setup [Non+19], where different data- and view-specific techniques have been developed together with technical experts from the Fraunhofer IGD in Rostock. The implemented prototype was thereby based on a touch-table scenario for the Fraunhofer *Plant@Hand3D* framework [Aeh+13] that displayed different data attributes of a factory hall on a 55 inch multi-touch table to monitor and control machine information in a single-user, non-collaborative digital twin. Together with medical experts, this idea was adopted for the hospitalization of patients to handle the growing quantity and quality of individual systems in the healthcare sector, such as the clinical information system (CIS), the picture archiving and communication system (PACS), the radiological or laboratory information systems (RIS/LIS), or the electronic health record systems (EHR) [Sch+20]. Due to the ongoing digitization in this area, the medical staff has to take on the role of data scientists in order to understand the correlations between multiple tools, observe changes in the data, and assure the well-being of multiple patients at the same time. In this regard, Health@Hand [Non+19] acted as the central monitoring tool for medical facilities providing a 3D-model of an intensive care unit (ICU) to emphasize the changes of real-time vital data. This digital twin of a hospital ward contained different types of information about patients, rooms, staff, or inventory. These categories are defined as *data sources* that each holds a set of *elements* to be handled by the *data manager* [Non+19]. An example of this would be the data source patient that holds elements such as pulse, heart rate variability, or respiration rate.

The goal for the medical experts was thereby to observe and examine the state of multiple patients mediated by their incoming vital data to define and investigate a particular scenario for the detection of cardiovascular anomalies. This included heart rate variability, blood pressure, breathing rate, and, in some cases, blood sugar levels. As shown in Figure 6.1, their application stretches from the monitoring of the entire ward to the specific analysis of a particular patient.

In this way, several VATs from different systems were connected and integrated into one toolchain [Sch+19b]. This allows medical professionals to monitor and analyze the health status of patients without having to switch between different tools. Based on the available hospitalization systems, own implementations of said systems have been included in Health@Hand. Important examples of this are:

- An *eHealth Record Browser* for checking on the patient's general information such as name, age, and prior diagnosis in the EHR,
- An *Image Viewer* for inspecting images from medical examinations in the PACS suite,
- A *Shimmer Monitor* for accessing activity sensor readings from a sensor that captures real-time data of the patient's vital parameters and determines the estimated activity,
- A *Vital Data (VD) Dashboard* that displays a patient's vital data such as pulse, heart rate variability, or respiration rate,
- A *Cardio Anomaly (CA) Detector* for analyzing heart rate data, and

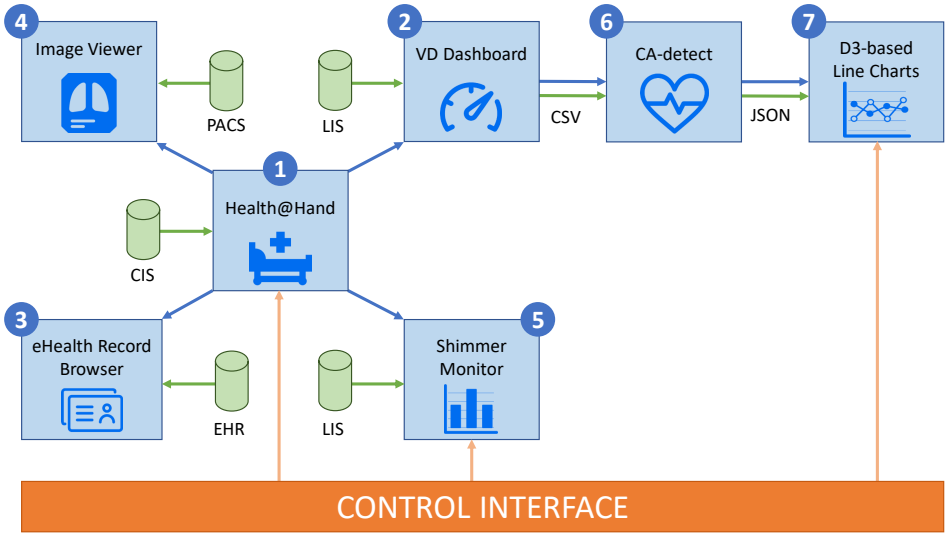

Figure 6.1: Combined view of a coordination model and the respective UI setup for the technique-centered evaluation of a medical analysis scenario on detecting cardiovascular symptoms and anomalies within a patient's historical and current clinical data utilizing multiple interconnected VATs [Sch+19b]. The usage flow is indicated by blue arrows, the data flow by green arrows, and the control flow by orange arrows.

- A *custom D3-based Line Charting tool* for visualizing the output from *CA Detector* and filtering potential abnormalities to point out reasons for a patient's irregular blood circulation.

On top of these are individual tools provided by sensor manufacturers, third-party analysis and visualization tools, and customized information dashboards. The data generated by each of these individual VATs is produced in real-time and guaranteed to be updated in intervals of ≤ 3 seconds. Through the central overview mechanic of the Healt@Hand user interface, the medical staff can review the information on screen to (1) check for irregularities and (2) investigate possible causes for them and, if necessary, call in the appropriate specialist. While these tasks might sound simple at first, there is a need to go through several steps for this due to the medical correlations of different vital parameters.

6.1.1 Specifying the Usage Flow

For the *temporal order* within the toolchain, seven reoccurring steps have been identified to check for irregularities (steps 1-5) and investigate possible causes (steps 6-7):

1. **Situational assessment using Health@Hand:** The medical expert monitors the general condition and state of assigned patients. From this overview display, the expert can select patients with critical vital data for further examination.

2. **Vital data overview using *VD Dashboard*:** A backlog of a patient's vital signs can be examined through diagrams and gauges in this dashboard. For more context, a patient's diagnosis and treatment plan can be opened up from the Health@Hand overview.

3. **Diagnosis details using *eHealth Record Browser*:** From the patient records, the entries *age* and *diagnosis* help the medical expert to put the vital signs in context. An additional display of the outcome of any procedures can be triggered from the Health@Hand overview as well.

4. **Checking imaging data using the *Image Viewer*:** The medical expert checks available imaging data relating to the cardiovascular system, as it is the heart rate showing irregularities. As both pulse and respiration correlate with the physical activity being performed, the patient's activity data can be brought up from the Health@Hand overview.

5. **Activity analysis using the *Shimmer Monitor*:** Looking at the patient's recent activities, including any therapeutic stressing situations, the expert finds no natural cause for the current irregularities. He thus switches back to the *VD Dashboard* to run an automated anomaly detection.

6. **Automated anomaly detection using *CA Detector*:** The cardiac anomaly detection calculates anomaly scores on the vital data streams. After some minor configuration, the results are automatically opened in a line chart.

7. **Anomaly analysis using the *D3-based Line charting* tool:** The medical expert inspects the identified anomalous spots and annotates sections that point towards reasons for the patient's apparent irregular blood circulation.

6.1.2 Specifying the Data Flow

For providing a common *data channel*, the individual VATs have been integrated with the help of different interfaces to pass available data between VATs consist of two parts: (1) the general medical background information about a patient and (2) the situation-specific data that is currently observed and requires analysis. While the medical background information is available from a number of centralized systems within the hospital and can be queried via the patient's identifier (ID), the situation-specific data is only available locally. In order to achieve both, a two-step procedure was employed.

For accessing medical background information , the patient's ID is passed as a parameter into the tools, which then query the relevant data records themselves. This variant of centralized data access is employed by tools such as the *eHealth Record Browser*, the *Image Viewer*, and the *VD Dashboard*.

For the situation-specific information , data is passed directly from one VAT to another, requiring a full-fledged data channel between them. The variant of decentralized data access is employed by tools such as *CA detector* and the *D3-based*

Line Charting. The passing of this data is denoted in Figure 6.1 by a second arrow in green connecting the respective tools.

These first steps were extended in a related bachelor thesis [Zie19] by a so-called "toolchain view" (see Figure 5.6), which provides an overview of the analytical toolchain graph, records all performed operations and enables the user to carry out analysis processes. In a follow-up master thesis [Zie20], this was enhanced to also allow data exchange with non-integrated VATs such as Microsoft Excel so that their results could be linked across multiple views.

6.1.3 Specifying the Control Flow

For the specification of *coordination rules*, adjustments between all time-oriented data displays have been investigated. The usage flow (see Subsection 6.1.1) shows that tools are mainly applied subsequently as none of the workflow steps requires using multiple tools at once.

Nevertheless, that does not mean that multiple tools can not be left open on the large touch screen for further reference – e.g., the *Shimmer Monitor* providing the overview of a patient's activities – or easier adjustment of input data – e.g., the *VD Dashboard* for selecting different time intervals of interest to be processed by *CA detector* and its results being updated in the *D3-based Line Charting* tool. As independent as these analysis steps and each respective tool come, they share parameters that benefit from being controlled globally through a central interface. An example of such a parameter occurs when the medical expert switches between hourly, daily, and weekly temporal resolutions to find earlier incidences on a larger time scale or if the medical expert looks into details of a found incident on a smaller time scale. The temporal resolution can also be adapted globally via a menu bar located at the bottom of the user interface, acting as a control hub for the entire tool ensemble.

6.1.4 Designing the UI for the Modeled Workflow

The UI design for the described scenario mirrors the workflow in most aspects. This is why the coordination model and UI setup have been integrated into a single schematic depiction shown in Figure 6.1 following the star topology discussed in Subsection 4.3.3 with the digital twin depicting the role of the central VAT. This choice was made to tie-together the UIs of the individual VATs on top of an inherently spatial digital twin of the medical facility to provide sense of locality for the UIs, linking them to the respective patient in question. This can be seen in the screenshot in Figure 5.6, where Health@Hand is shown in the background of the UI ensemble and other tools are opened on top of it.

The control interface was designed to take the form of an unobtrusive menu bar at the bottom of the screen that is always visible and gives access to functionality for managing and parametrizing the UIs. The positioning and appearance of the control interface furthers the impression of the UI ensemble as a desktop environment where tools are centrally managed through a taskbar. By mimicking the desktop metaphor, users feel

comfortable and knowledgeable about managing the ensemble without much training. On top of the familiar visual appearance, the functionality of this environment is carefully adjusted to support coordination along the workflow. For example, workflow-sensitive taps open up the next tool in the workflow with every tap on the same patient – i.e., the first tap opens the *VD Dashboard*, another tap opens the *eHealth Record Browser* and so on.

6.1.5 Conclusion

Through the initial approach for lightweight tool coordination between multiple VATs [Sch+19b], we were able to establish centralized, unified access to otherwise loosely coupled, decentralized VAT ensembles within the Health@Hand framework. The proposed decentralized approach captures coordination in VAT ensembles by pairing tools together with a collection of prototypical UI setups. Since each analysis scenario brings its own execution sequences, this demonstrated the proposed design process, coordination model, and UI setups using a health analysis application example. The freedom of configuration in Health@Hand enabled us to identify potential problems based on the application scenario. This way, it was possible to define four initial interface ensembles as feasible solutions in the subsequent work.

By testing different approaches with the demonstrative example of Health@Hand, it was possible to gather a lot of feedback from technical experts as well as domain users from the medical field. Health@Hand and its lightweight coordination capabilities have been presented at the Medica World Forum for Medicine. The user feedback had been very encouraging, including comments such as "That is the future of clinical data exploration." (from *PAMB*) or "The integration of different data views from different tools will improve diagnosis and therapy management" (from *Poly-Projekt GmbH*). Especially, changes in the VATs itself and the visual connections between them in the overview UI proved to be very helpful for the users.

6.2 Workflow-centered Evaluation

At the stage of workflow-centered evaluation, a necessarily complex toolchain with independent existing VATs was built in order to showcase and gather feedback about the implementation (see Chapter 5). Rather than coordinating VATs of a toolchain within one system like in Section 6.1, this stage focused on connecting independent VATs of a workflow based on the first prototypical setup of a *AnyProc-ReVize* setup [Non+22]. For this, a reasonable subset of data-driven exchange mechanisms was selected together with visualization experts from the Aarhus University in Denmark.

For the specific workflow, the data from the critical care dataset MIMIC-III [Joh+16] was used, which contains about 46,000 patient records, 58,000 hospital admissions, as well as 17.5 million registered fluid intake events and 3.6 million fluid output events. Based on the introduced role system (see Subsection 3.3.2), we assumed a physician or a hospital administrator to be the domain expert, while a data scientist or health IT

professional would act as the technical expert to configure the toolchain.

6.2.1 Specifying the Usage Flow

In order to justify the independence of this approach, a diverse set of VATs was used to analyze the MIMIC-III dataset [Joh+16], allowing for a back and forth between them if needed.

KNIME [Ber+08b; Ber+08a] is a powerful monolithic VAT with various computational analysis capabilities. In the context of this approach, it was used for preprocessing and data preparation to filter and select relevant information using unsupervised machine learning methods like clustering and PCA. A disadvantage of KNIME, however, is its limited visual-interactive properties, which hamper the exploration of the computed results.

VisFlow [YS17] is a VAT with the necessary properties for the flexible configuration and exploration of data visualizations. In the context of this approach, it was used for the visual display and interactive analysis of the extracted data subsets. However, Visflow clearly lacks the advanced computational capabilities of KNIME and provides a lot of freedom to non-visualization authors.

ColorBrewer [HB03] allows to specifically constrain the individual design choice to pre-defined color palettes (e.g., color-blind friendly palettes). In the context of this approach, it was used to adjust and fine-tune color scales in the plots generated by VisFlow. While ColorBrewer cannot be used interactively or for data preparations, it still enhances the functionality of the other VATs.

Figure 6.2 illustrates the usage flow among the three VATs that each excel at a different stage of the analysis process.

6.2.2 Specifying the Data Flow

The technique-centered view on the coupling of tools formed the basis for a data-flow-oriented model, followed by defining how the three VATs in this scenario pass data among each other. This requires some code changes, as, for example, ColorBrewer, in its original form, does not allow any data to be passed in or out. However, reciting the challenges of this thesis (see Subsection 3.1.2), the concept of this thesis demanded less coding effort than integrating ColorBrewer's functionality into VisFlow. Since both tools are open source and web-based, the ReVize web sockets were chosen as a communication interface (see Chapter 5) between tools, such that they exchange a Vega-Lite specification of the latest visualization with the next tool in the toolchain through web sockets. To achieve this, variants of VisFlow and ColorBrewer and KNIME have been used that are "ReVize-enabled", meaning they have been extended to support this channel.

By doing so, it was possible to configure the connections as needed (as shown in Figure 5.2), which results in the following data exchange characteristics [Non+20]: Each

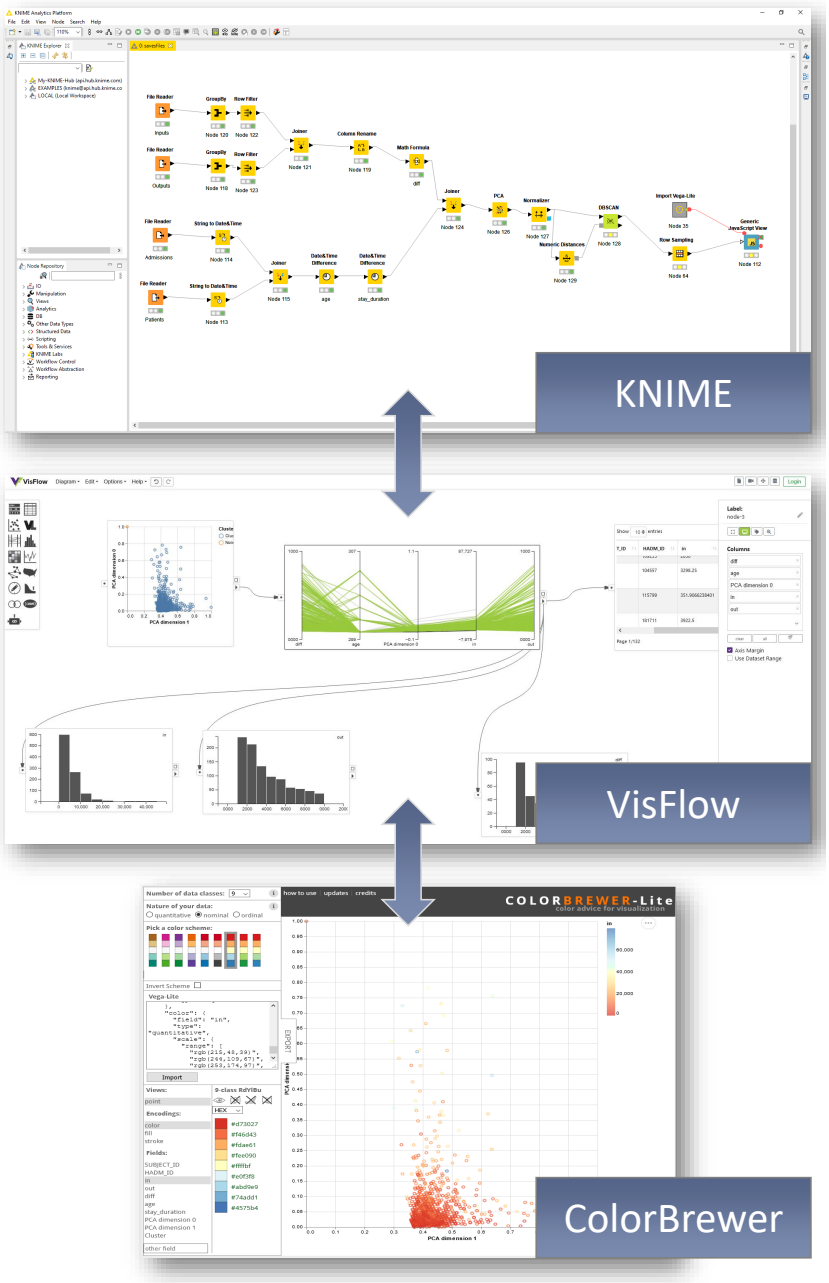

Figure 6.2: Screenshots of KNIME, VisFlow, and the ColorBrewer from different points during the analysis process for the workflow-centered evaluation. The information displayed has been exchanged between the programs and refined in the context of every task within the toolchain.

data transfer includes the *full dataset* together with additional *metadata about its visual representation* using the *structured data format* Vega-Lite. Using the ReVize server as a *central infrastructure*, the data exchange is used for *inputs and outputs* from tools, as well as to distribute interactive *modifications* from *one to many* other VATs connected to that server. This is done *bidirectionally* (backward and forward along the toolchain) in a *synchronous* fashion where changes are *pushed* to the server when switching from one VAT to another. At all times, the connected VATs have *full access* to the most recent Vega-Lite specification, which is kept *persistent* on the server.

6.2.3 Specifying the Control Flow

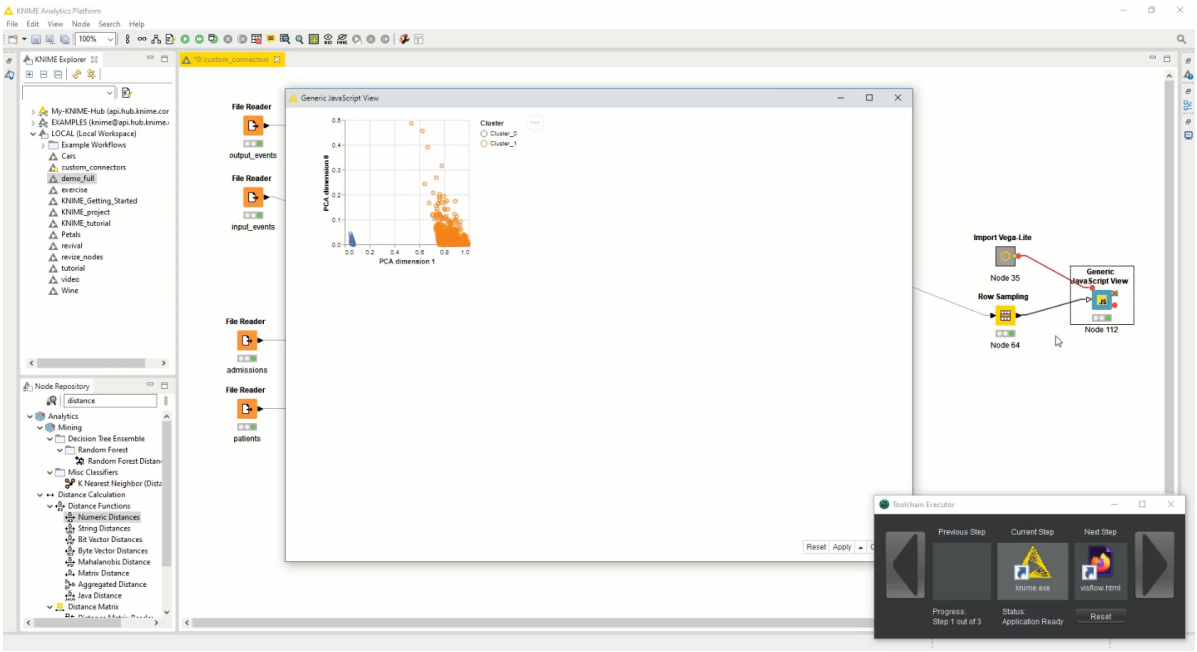

Figure 6.3: Screenshot showing the visual output from the initial step in the application scenario. The AnyProc executor module (in the foreground on the bottom-right) overlays the local KNIME installation (in the background) to help the domain expert in navigating the pre-defined toolchain.

1. KNIME is started as the first VAT in the toolchain.

2. Since it is configured to connect via ReVize to the next tool in the toolchain, KNIME is already opened with a minimal analytic pipeline showing the four data sources and a final Vega-Lite node that will output the necessary specification.

3. Within KNIME, data transformations (e.g., date conversions), computations (e.g., determining the patients' ages), cleaning (e.g., removing missing entries), and aggregations (e.g., summing up fluid intakes and outputs per admission) are added before joining all data of interest into a single table.

4. Next, a PCA will be conducted on this multi-dimensional data to project it into two dimensions.

5. In addition to this mapping, a DBSCAN clustering analysis is performed to identify groups of patients, and the resulting data is plotted as shown in Figure 6.3.

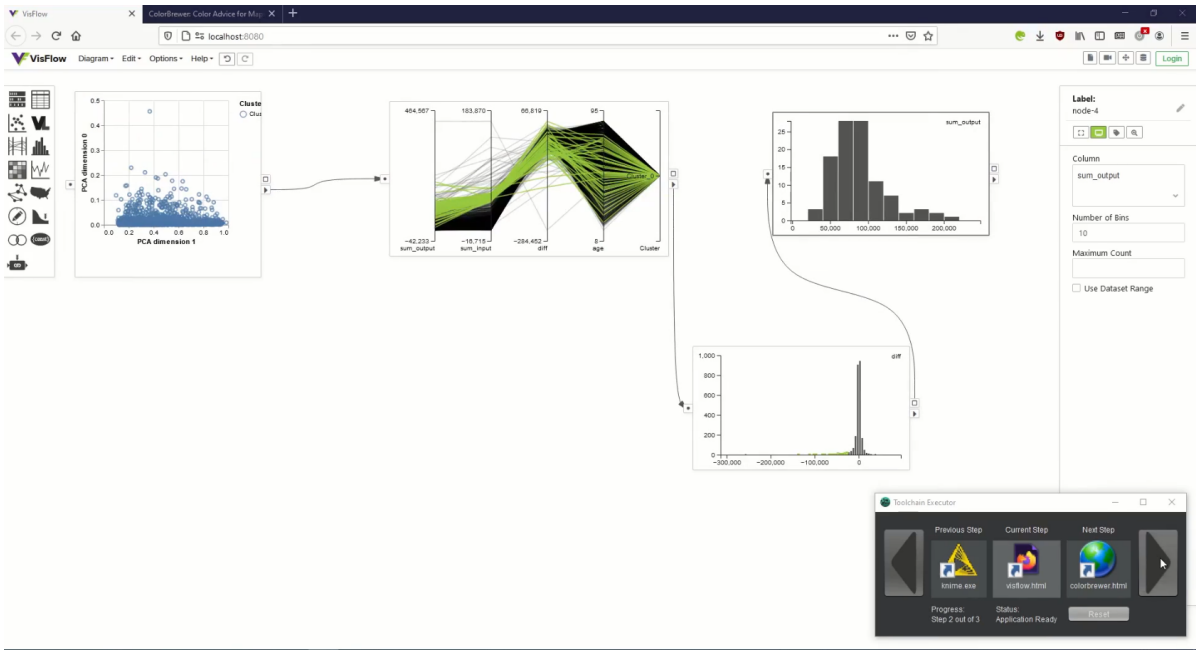

Figure 6.4: Screenshot displaying the visual result from the VisFlow web tool (in the background) after filtering out incorrect information with KNIME. Switching between these two VATs is possible through the AnyProc executor module (in the foreground in the bottom right corner).

6. Switch to VisFlow for interactively exploring the plot. To do so, the [Next] button in the AnyProc executor is pressed, which opens up VisFlow with the plot generated in KNIME.

7. In the background, KNIME has continuously pushed any updates of the plot to the ReVize server – but only now, with the click of the button, the ReVize server is instructed to pass these updates on to other tools. Furthermore, since the Vega-Lite grammar does not capture all transformation operations performed in KNIME, the results of the transformation are instead added to the data as additional columns. All of this is hidden from the domain expert, who only sees VisFlow starting up and showing the plot from KNIME.

8. Now VisFlow is used to add parallel coordinates that provide more detail on the cluster characteristics.

9. Then brushing & linking and interactive tooltips are applied to explore the clustered data. Through this, we soon realized that one of the clusters contains patients that

are over 300 years old. According to the documentation of the dataset, this is an artifact of the data anonymization process.

10. To prevent these unusually old patients from skewing subsequent analyses, one should go back to KNIME to filter out these data. A click on the [Previous] button in the AnyProc executor brings up KNIME again.

11. Revising the analysis pipeline to filter out patients older than 130 years. After re-running the pipeline, the cluster of patients over 300 years old is gone.

12. Via the [Next] button, we switch again to VisFlow, relying on AnyProc to update the contents in VisFlow according to the changes made in KNIME.

13. In VisFlow, the data is explored further using the particular workflow node that is available in the Vega-Lite enabled version, which can render, send, and receive Vega-Lite specifications from the ReVize server.

14. We find some curious cases of patients who were given medication but whose fluid intake is nevertheless registered as 0 ml or even −1 ml. On top of that, we notice that the absolute fluid levels are not very useful for the analysis as they vary drastically with the duration of the hospital stay. Instead, we would like to see, for example, a patient's average fluid levels per day or the difference between a patient's overall fluid intake and output. Therefore, the process is switched back to KNIME using the [Previous] button, the necessary filters and computations are added to the pipeline, and the PCA is adjusted to use the newly computed values.

15. Switching to VisFlow via the [Next] button updates the charts in VisFlow (shown in Figure 6.4). When investigating the newly computed difference between fluid intake and output, we would like to use a diverging color scale to better discern net loss from net gain. Unfortunately, VisFlow only offers a limited set of color schemes to choose from, and ensuring color-blind friendliness is not possible, either. To nevertheless make the desired changes to the charts' colors, we switch to the next tool in the toolchain using the [Next] button.

16. This opens ColorBrewer, showing the visualization from the Vega-Lite node in VisFlow. As was the case for KNIME, any update in VisFlow was already constantly pushed to the ReVize server in the background – the button click then triggered the server to distribute the latest update to the connected tools. Now, one can select an appropriate color scale while previewing its effects on the visualization (see Figure 6.5). When done, it is time to go back to VisFlow using the [Previous] button, and there, the changes made in ColorBrewer show up. Using VisFlow, we find more implausible data, such as patients who output an average of 17 liters of fluid per day or patients who output over 160 liters of fluid more than they took in. Removing them by adding more filters to the preprocessing in additional roundtrips between KNIME and VisFlow would be a plausible option at this point. Yet, this

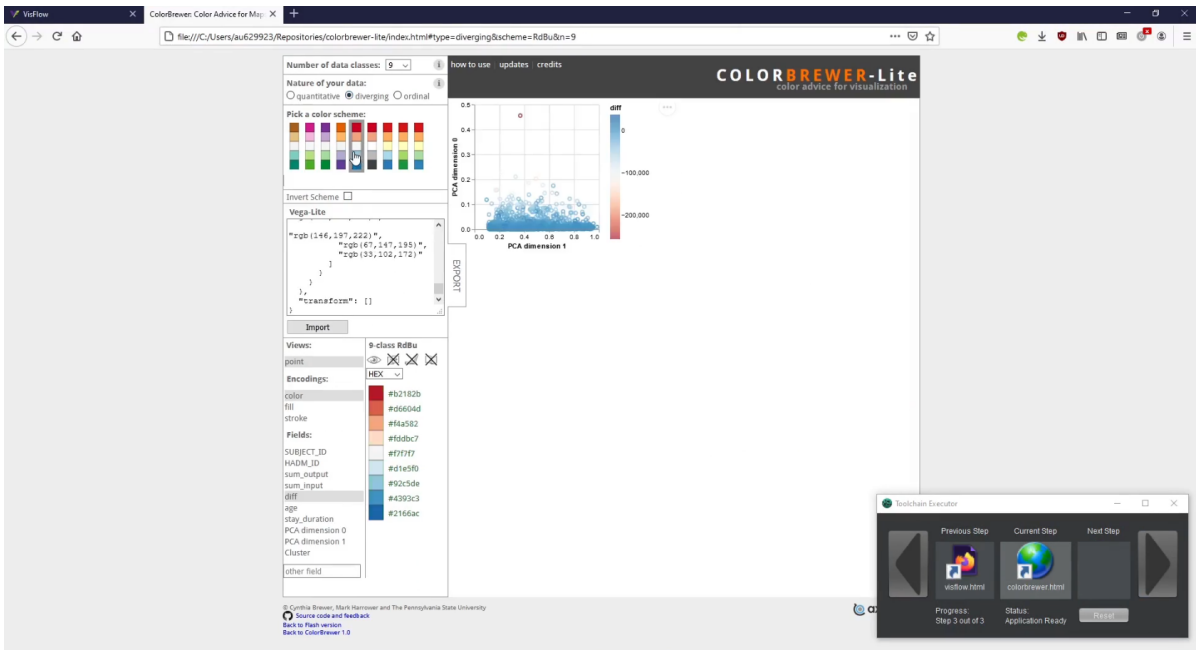

Figure 6.5: Screenshot showing the visual result from the ColorBrewer (in the background) after applying a new color scale for the VisFlow visualization. The AnyProc executor module (at the bottom-right) allows the user to confirm these changes and return to the VisFlow tool.

no longer seems like a viable solution. Instead we opt for adding the data cleaning tool OpenRefine [Huy21] to the beginning of the toolchain.

Up to this point, we made seven switches between VATs that would usually involve navigating the file system to export and import the data, as well as navigating the start menu to find the next VAT to be used. This is, of course, assuming that the tools rely on the same file format.

6.2.4 Customizing the Usage Flow and the Data Flow

Integrating a new tool like OpenRefine into the analysis workflow during the execution is an essential feature for accommodating the exploratory nature of many analyses and delving into the unexpected insights they may produce. To that end, the usage flow was extended by opening the AnyProc editor and adding OpenRefine as a new VAT into the toolchain. After positioning it at the start of the chain, it was necessary to redirect the data sources from KNIME to OpenRefine. Finally, we had to specify how it connects to KNIME, determining the data channel for output transfer. Since OpenRefine is incompatible with the ReVize server, we add a connection node (shown in Figure 5.4) to change the data channel to the local file system. With OpenRefine in place, the user has now access to a powerful suite of cleaning functionalities, which makes it much easier to find and mitigate inconsistencies in the input data. For example, it is possible to

identify inconsistencies along entire columns and rows and remove those entries from the dataset with a few clicks. This way, we are able to filter the MIMIC data tables prior to their preprocessing in KNIME.

6.2.5 Conclusion

The first prototypical implementation of AnyProc simplified the configuration and execution of domain tasks under the umbrella of a unified interface [Non+21]. While the costs of switching between VATs have not vanished, they are made transparent and paid once up-front when setting up the usage flow and data flow so that the technicalities involved no longer interrupt the analysis. While the presented scenario itself is not particularly complex, it still requires a significant amount of back-and-forth between different VATs. In that sense, it is able to show the fundamental challenges of a cross-tool analysis and how domain expert are struggling with it.

As shown in Subsection 6.2.3, already a rather simple setup of two or three tools can make a considerable number of tool switches necessary. The more switches are necessary in an analysis, the larger the benefit of using AnyProc becomes. Through AnyProc, the workflow-centric evaluation of independent VATs can be applied to various application areas, such as the analysis of retinal image data, which is described in the following Section 6.3. Overall, the conceptual approach of workflow-centered VAT coordination received very positive feedback from the scientific community and was later enhanced in an extended version to further specify details and add additional functionalities for a more flexible configuration [Non+22].

6.3 Application-centered Evaluation

At the stage of application-centered evaluation, multiple tools and data sources required in ophthalmic research were included in collaboration with domain experts from the medical faculty at the University of Rostock. Rather than testing developed coordination mechanisms in isolation or combining individually chosen VATs, this evaluation phase went further ahead toward a practical usage of the presented prototype by comparing it to task-relevant toolchains in the medical domain. The result of this is a *ophthalmology setup* shown in Figure 6.6, where we defined a use case involving workflow-based visual analysis of retinal data [Röh+19]. For this, interviews with ophthalmologists and job observations were conducted to get an overall idea of the current analysis practices. Together, we engaged in the statistical analysis of two studies [Göt+18; Pra+18] and subsequently two additional studies [Pra+19; Pra+20] analyzed with a mixture of statistical and VATs. This provided deeper insights into the domain experts' workflows and use of VATs, statistics, and general-purpose analysis software, and it ended up in a list of design requirements related to the coordination of VATs for retinal data analysis.

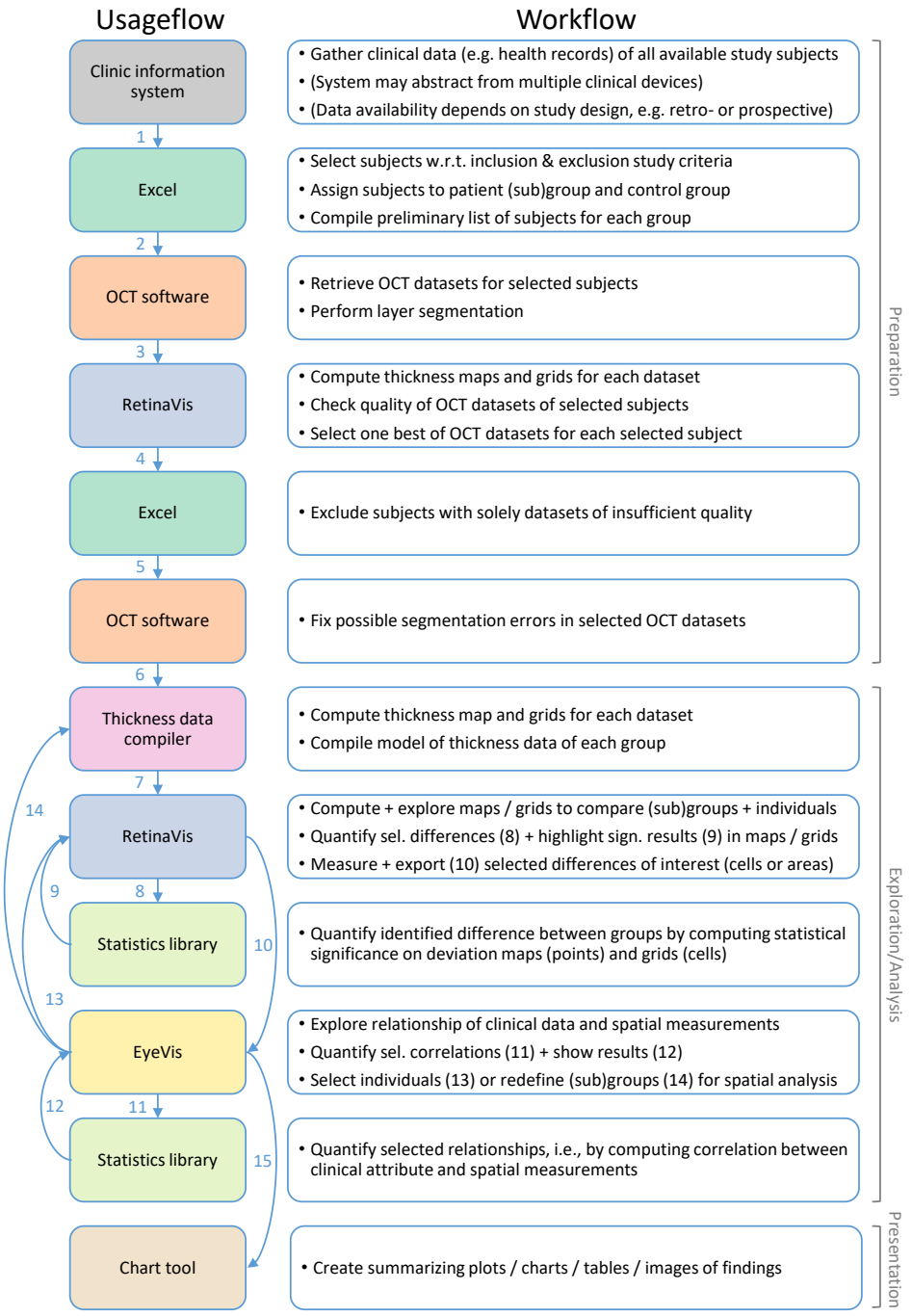

Figure 6.6: Visual representation of the complex usage flow for the optical coherence tomography that is extracted from the information given by the tasks that define the workflow of the domain experts.

6.3.1 Specifying the Usage Flow

The ophthalmologists were particularly interested in analyzing retinal image data in the context of cross-sectional studies. They wanted to study the retinal changes in relation to healthy controls of patients. By gathering retinal 3D image data acquired via optical coherence tomography (OCT) and other clinical parameters of the two study groups from a clinical data management system. The data analysis was then carried out in three main steps:

1. Data preparation: Compilation of study groups and data quality checks.
2. Data exploration: Discovering differences between study groups and investigating relationships with clinical parameters.
3. Data presentation: Summarizing findings and reporting study results.

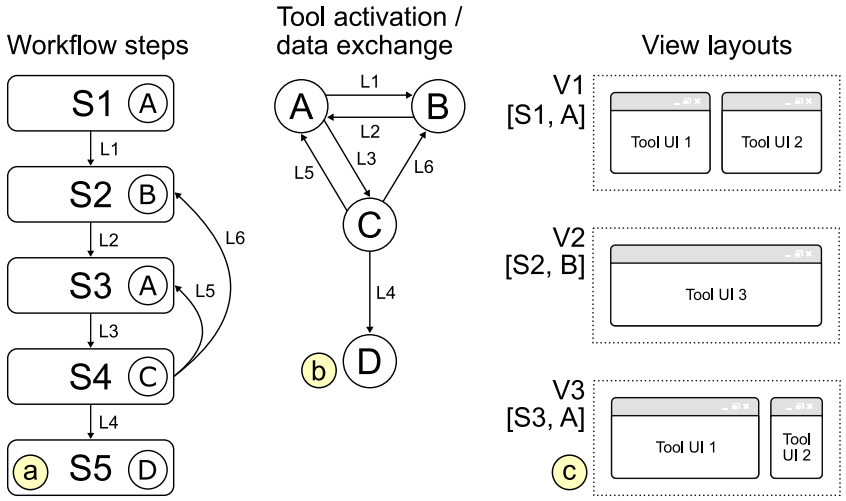

Figure 6.7: Overview of the coordination components for the application-centered evaluation with workflow steps *S1* to *S5* (a), associated toolsets *A* to *D* (b), links between workflow steps and tools *L1* to *L6*, and view layouts *V1* to *V3* assigned to each workflow steps and toolset (c).

Each of these parts came with several sub-tasks in order to compare patient images with normative data and other measured clinical parameters. In this process, a set of eight diverse tools was applied. This included commercial software, e.g., device-specific OCT software and spreadsheet software, as well as statistics software and visual analysis software, e.g., the R software environment [Tea] and a custom visual analysis framework for retinal OCT data [Röh+18; Pra+19]. An overview of all coordination components can be seen in Figure 6.7. In order to define a suitable usage flow between these tools, we discussed objectives, explained possible analysis variations, and jointly identified challenges for the automated support with their current use of independent VATs without any kind of automated support for activating tools as needed [Röh+19]. Together with

the domain experts, the content of the tool container, data source container, and graph container of the editor was defined in the following three steps:

1. learning which tools and data are required by taking a closer look at the retinal data analysis,
2. temporally connecting the tools by tracing the order in which they were applied in the experts' analysis procedure and
3. assessing the benefits of this first layer of tool coupling by gathering informal feedback and reflecting on results.

While AnyProc's visual presentation of temporal connections aided in getting an idea of when which tool was applied, the direct drag & drop manipulation support helped to quickly refine the initial sequence based on the experts' feedback and devise alternatives.

Through this, the content of the tool container, data source container, and graph container were defined by (1) learning which VATs and data sources are required, (2) temporally connecting the tools by tracing the order in which they were applied in the experts' analysis procedure and (3) assessing the benefits of VAT coordination through gathering informal feedback and reflecting on results.

6.3.2 Specifying the Data Flow

The data flow configuration aimed to streamline data transition between tools by modeling dependencies and the order in which data moves across steps. The data required for retinal analysis included two primary raw sources: OCT scans and clinical health records. While OCT scans provided detailed retinal images, health records included parameters like age, blood glucose, body mass index, and medical history of patients. By integrating these data types, the experts could analyze structural changes in retinal layers and correlate these findings with clinical characteristics.

In the editor's graph container, the experts set up pathways for data to transition between tools. Data connections ensured that results from one tool, such as calculated retinal layer thickness, could be directly used in the next tool, for example, for statistical correlation analysis with clinical parameters.

The main objective was to reduce the redundancy in data loading across tools and avoid inconsistencies. This was achieved by creating a unified data access point, where data was stored in a standard format compatible with each VAT. The visual representation (illustrated in Figure 6.8) displays how OCT scan data and clinical parameters flow through the analysis stages. For example, raw OCT data first passes through an OCT processing tool that prepares segmented images, which then move to VATs specifically designed for statistical and visual analysis.

The experts also tested data flow pathways to validate the sequence of data transitions and correct formatting issues between tools. This step was crucial for ensuring the model's reliability when transitioning between tools with varying data requirements or formats. The refined data flow, therefore, minimized disruptions during the actual analysis and allowed for a smooth, continuous flow of data between workflow stages.

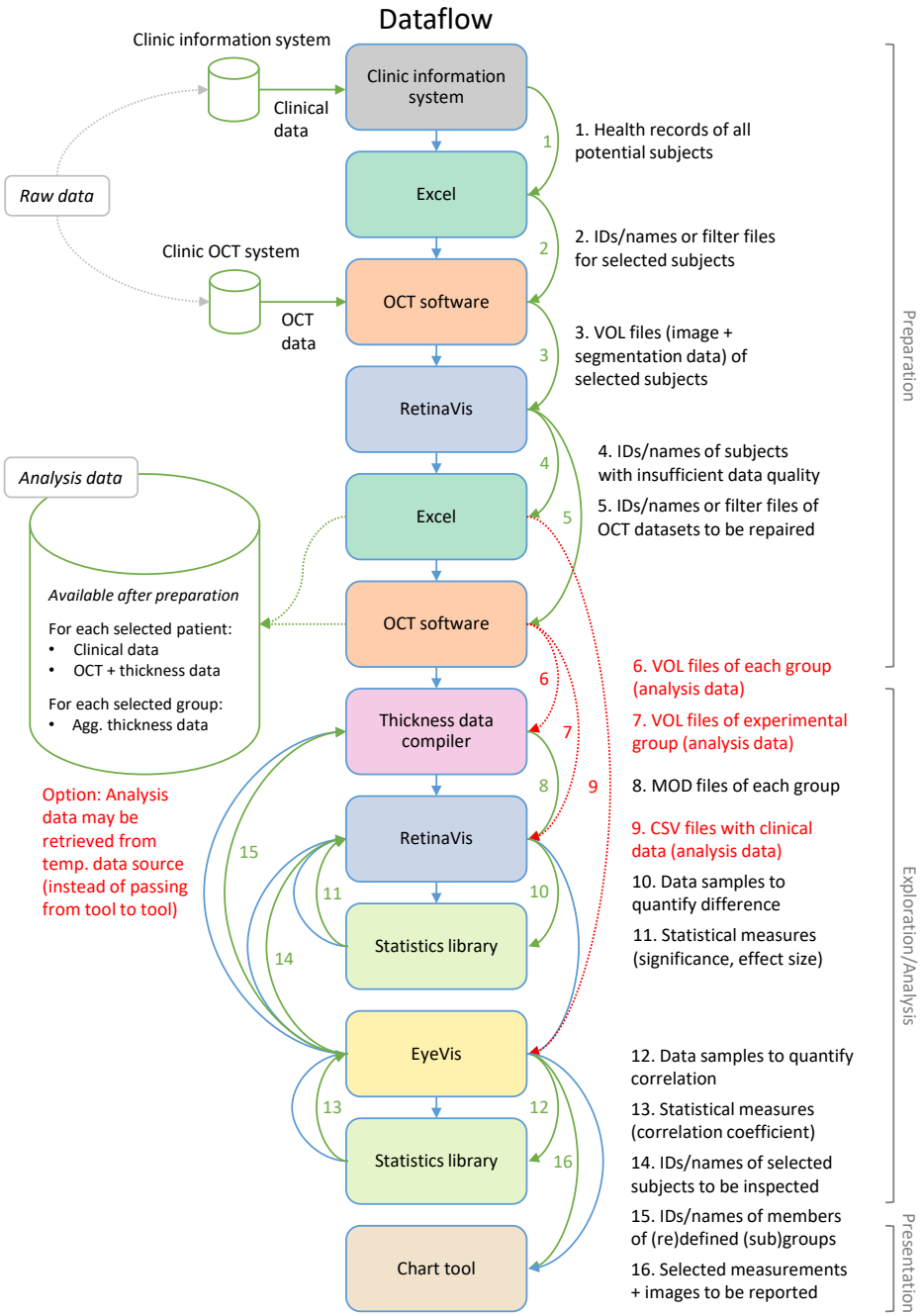

Figure 6.8: Visual representation of the data flow for the optical coherence tomography based on the extracted information of the usage flow to determine the coordination between the visual analytics tools with the underlying data sources.

6.3.3 Specifying the Control Flow

To amend the data and support in locating patterns and possibly generate findings, suitable visualization, interaction, and annotation techniques were provided to the domain experts. This enabled the ophthalmic experts to analyze the dependency between treatment and visual acuity development and showed that the integration of suitable workflows with visual analytics methods can be beneficial for task fulfillment in clinical environments.

The control flow addressed the navigation of tools, adjustment of parameters, and annotation of findings. For each VAT, customized interaction options were defined, specifying which tools to activate, how parameters should be adjusted, and how data from prior analyses could influence the current tool's settings. Figure 6.9 shows a schematic of the expected control flow, with each step indicating user actions, such as selecting and computing statistical measures, removing subjects of insufficient quality, or filtering and segmenting layers in datasets.

Although most data funneling and parameter synchronization still had to be done by hand, the modeled VAT sequence and automated VAT activation already helped them to focus on the actual data processing and analysis.

6.3.4 Conclusion

Through AnyProc, a VAT sequence was built on top of an established scientific analysis procedure for retinal data in ophthalmic research. The editor proved to be a useful to successfully model a VAT sequence on top of the analysis procedure explained by the ophthalmic experts. In fact, throughout this process, other ophthalmic analysis procedures similar to the described use case were identified, including studying specific diseases or investigating corneal study data. In this context, the editor would support defining new temporal VAT sequences and help to revisit and adapt existing ones. The latter becomes especially necessary if medical devices and respective software are changed in a clinical environment or if additional steps are added on top of the established analysis procedures. By serializing and storing a modeled VAT sequence, it is even possible to go back to previous iterations or to recreate a specific analysis result.

Regarding the temporal coordination of VATs, the experts were relieved from the time-consuming tasks of having to repeatedly search for each required VAT and starting it independently. Especially moving between different analysis tasks and frequently switching between various tools demanded significant manual effort. The structured guidance throughout the analysis process ensured that no VAT or critical steps, such as data quality checks and necessary corrections, were accidentally overlooked. Moreover, it enhanced the comparability of different analysis runs by consistently activating VATs in alignment with the experts' established procedures.

Comparing the execution of the workflow supported by the unified UI [Röh+23] with the domain experience of manual tool coordination showed that the implemented approach reduced the effort for the experts and allowed them to recapitulate the steps and intermediate results. They were able to assemble the tools on a timeline, add data sources

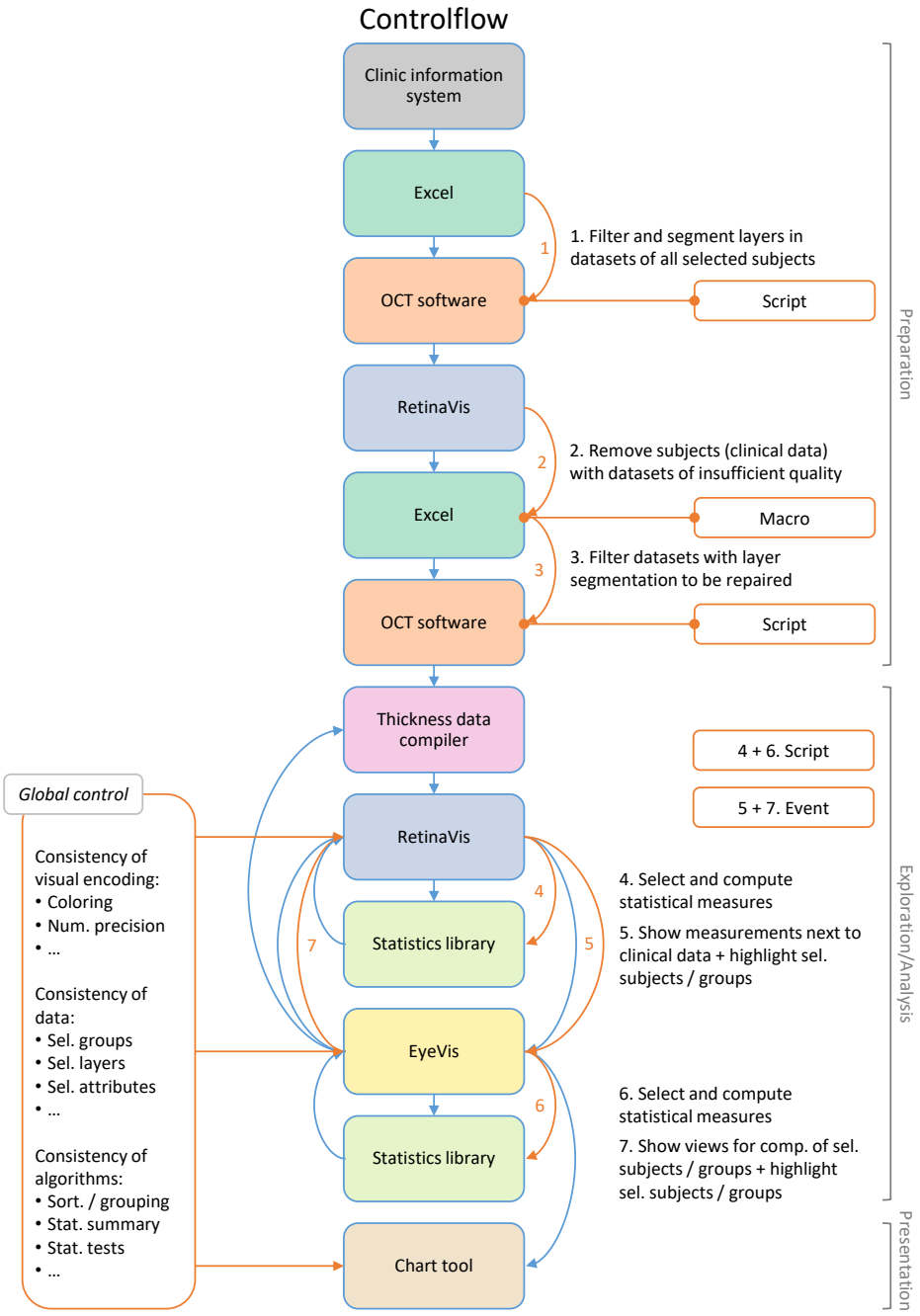

Figure 6.9: Visual representation of the control flow for the optical coherence tomography based on the expected interaction from domain experts with the pre-configured toolchain.

and transformations, and connect the work steps to the resulting coordination graph. The unified UI allowed them to walk through the steps as the tools were activated and their views automatically arranged. One example is the acknowledgment of the commonness of extraordinary incidents that, while unrelated to the initial treatment, still impact visual acuity development. By supporting the externalization of data modifications and discoveries, we enable continuous data refinement and the possibility of generating new insights.

Applying the visualization system together with the experts triggered discussions and sparked new ideas for further improvements, such as the task-based composition of views. Other directions for future work included the dynamic rearrangement of views based on user focus during the analysis, task-based composition of view parts into combined interfaces, and extended global parameter control. Yet, given the heterogeneity of tools and data regarding this specific use case, this would have involved a very high coordination effort that requires further studies.

7 Discussion

In this chapter, the results of this work are compared with the original research questions to discuss if the provided concept, implementation, and evaluation resemble the originally intended goal of this research. This not only highlights the insights gained throughout this research but also argues limitations in order to present possible starting points for future improvements.

7.1 Finding Principles Types of Data Exchange

In order to answer which principle types of data exchange between VATs exist, a typology of data-level coordination mechanisms between VATs was constructed that abstracts from the many different technical realizations by grouping them according to their characteristics.

For this, a systematical investigation on various existing VA systems such as KNIME [Ber+08b], Metadata Mapper [RM11] or VisTrails [Mor+13] was performed to define how data exchange between independent VATs can be realized. The goal was to find those characteristics that are actually relevant for coordinating VATs and to associate them with the defined coordination graph. The resulting taxonomy in Section 4.2 describes major characteristics of data exchange as *data*, *function*, *topology*, *chronology* (when is data exchanged?), and *availability*.

During the technique-centered evaluation (see Section 6.1), it was detected that the various systems can differ significantly with respect to the pre-defined aspects relevant to VAT coordination. To examine this in more detail, the developed taxonomy was applied on three systems: Plant@Hand3D [ASU13], ReVize [HS19] and an experimental version of AnyProc. Through this, a correlation between the completeness of the definition and the flexibility was detected. While the more robust systems have been fully defined, the more flexible ones have some missing values for different aspects of the characterization. Yet, these gaps are by no means negative as they just show the clear correlation between the robustness of a specific solution and the flexibility of a general solution. The taxonomy can, therefore, be used to find criteria for further evaluation, comparison, and design adjustments by identifying aspects of data exchange that alter systems with additional functionality like version control, data transformations, or progressive analytics. The hypothesis that there are principal types of data exchange that can be defined for various VATs independent of the implementation-specific mechanisms can be verified through this model.

All in all, the taxonomy allows descriptions of the data exchange between VATs in a general and abstract way [Non+20]. It provides a descriptive vocabulary, which abstracts

from domain specifics and underlying technologies. Therefore, it can be seen as a first step to detect the "space of possible" in terms of data flow coordination and make sense of what already exists [KK17]. In conjunction with the lightweight coordination approach, this can be used to characterize data exchange and define VATs independently of their underlying structure. This helps systematize the design of visual analysis systems, following the previously established coordination mantra of "integrate what is necessary and couple what is possible."

7.2 Representing the User's Workflow

In order to answer, what should be shown to represent the user's workflow, the user interface must take into account the information that needs to be displayed at a given time and the context in which it is displayed. Regarding the information, two fundamental types have been identified during the technique-centered evaluation with the technical experts at Fraunhofer IGD Rostock (see Section 6.1): 1) the workflow with the current tools and 2) the data associated with these tools, including other details such as data transformations and parameter settings. In terms of context, two basic cases were derived from this: 1) when the workflow and toolchain must initially be configured by the user and 2) when an already configured workflow is executed during an analysis.

For the configuration, AnyProc was developed with a design that only provides the necessary information to create a toolchain for a specific workflow [Non+21]. The workflow is thereby represented as a chronological sequence of time slots, while available tools and data sources are visualized as thumbnails so that they can be easily recognized by the user. Intuitive interaction techniques were integrated into this editor component so that the user can easily place tools on time slots in the order they have to be activated, attach data sources to the tools, and add data transformations to specify the required connections in the toolchain. The resulting design shows information for a single time point and provides an overview of all configured time points. The toolchain can be run up to a selected time slot to fine-tune any parameter settings that govern the data exchange on the fly. This proved especially helpful when switching between testing and re-parameterizing partially configured toolchains to adapt the displayed information during the workflow-centered evaluation in Section 6.2. The configurations made through this user interface are used to update AnyProc's internal model of the coordination graph, which automatically determines what to show during the model's application.

When executing workflows, the user must first and foremost be provided with information about the work to be performed. We thus developed a new user interface design that focuses on the workflow steps and assigned tools [Röh+23]. Two user interface modes show either an overview of the steps to get a first idea of all the tasks in the workflow or a detailed view of the step currently being processed, including descriptions of the actions and tools involved, the status of the work, and links to previous and upcoming steps. Both modes feature suitable navigation functions that allow the user to jump back and forth between steps, trigger queries to the underlying configuration, and automatically activate tools and update the display with the associated data. On request, additional information

about currently active data transformations and parameter settings is retrieved and made interactively adjustable in case requirements are dynamically changing during workflow execution. This way, up-to-date information about the execution is always available. The hypothesis that a unified UI can be constructed independent of the number of VATs shown on screen can thereby be verified, as the visual configuration interface of AnyProc provides the means to realize this desired behavior.

By defining all types of information and contexts, it was possible to create a complementary user interface design that delivers all relevant information that needs to be displayed both during configuration and execution of workflows at any given point in time [Röh+23]. Through this, it is possible to represent a first draft of the user's mental model by using the knowledge of both technical and domain experts together in order to configure and execute complex toolchains. Hence, it simplifies the configuration to the "space of possible" in terms of existing tools and workflow tasks, following the previously established mantra "integrate what is necessary and couple what is possible".

7.3 Defining Parts of Data for the Exchange Process

In order to answer what parts of the data can or should be exchanged between VATs, it is firstly necessary to establish a mapping between selected data subsets and different coordination level types such as the characteristics defined in Section 7.1. To achieve this, a fine-grained data synchronization concept was created to take coordination requests together with temporal and capacity constraints as an input and define suitable data flow on top of the coordination graph as an output (if such a data flow exists) [Non+21]. This idea caters directly to the needs of the configured toolchain (represented in Section 7.4) by being able to take into account temporal dependencies modeled directly on the coordination graph as additional, pre-defined constraints.

To this cause, we collaborated with visualization experts at the University of Aarhus to build on top of the conceptual models for VAT coordination [Sch+20] and data exchange [Non+20]. To realize such a combination of independent VATs and demonstrate its feasibility in use, we firstly examined various methods for automated partitioning or unification of task-dependent information reaching from system-specific applications [Zie20] over grouping mechanisms [Beh21] to AI-based unification models [Wul21]. While these approaches provided interesting results for specific application domains, it was not possible to identify a generalized solution for automated data exchange of relevant subsets between undefined VAT combinations.

Hence, the idea of data-flow-oriented visualization software [Str+12] was followed, except that instead of composing individual modules of the same system into one data flow, individual VATs have been used and their data exchange was performed along that flow by utilizing the available means of pairwise communication. To this end, the editor module of AnyProc [Nona] was developed so that it enables technical experts to customize and connect different VATs and data subsets into a configurable representation. Using AnyProc's executor module, the domain experts can then execute their workflow with little to no interference by starting and stopping the corresponding VATs with

the underlying data [Non+21]. The inclusion of a ReVize web server [HS19] helped us to define the data flow between VATs by selecting distinctive data subsets and mapping them to the available data layer. The initial editor [Non+22] was enhanced by defining multiple channels for data exchange via file system, web server, or copy & paste mechanisms and provided multiple features to cover the desired characteristics from the taxonomy in order to establish a mapping between the used data subsets and different coordination level types. The use of this system was demonstrated as part of the workflow-centered evaluation in Section 6.2 to analyze patient fluids input and output within the large Mimic-III dataset [Joh+16] using the independent VATs Knime [Ber+08b], Visflow [YS17], ColorBrewer [HB03] and OpenRefine [Huy21]. The hypothesis that scalable data exchange can be achieved by relying on transitional input and output management can thereby be seen as partly verified since the presented approach carries out the transitional management while the user has yet to define it beforehand.

By defining the relevant data subsets and their exchange through transitional input-output management, it was possible to create a scalable data synchronization concept for coordinating VATs. This approach allows tools to share only the salient parts of the data necessary for the analysis at a given time, optimizing performance while minimizing redundant data transfers. The implementation of AnyProc and ReVize enabled the integration of diverse VATs and data flows, deeply referring to "couple what is possible" on a data level regarding the previously established coordination mantra.

7.4 Structuring the Visual Representation of Analytical Processes

In order to answer how the workflow can be represented in a meaningful way, a unified user interface requires access to the visualization functionality of individual VATs as well as an overarching structure that visualizes 1) the data, 2) the workflow, 3) the tools, and 4) any associations between them based on the structure of the underlying coordination graph.

Following the established flow-oriented model for lightweight coordination [Sch+19b], flexible user interface setups have been defined to support the combination of all components to be visualized. These setups deliver suitable layouts for multiple tool views to integrate the individual user interface designs mentioned in Section 7.2 into a unified interface to display the workflow steps and associated tools along with the data and parameters encapsulated by the corresponding toolchain. This consisted of two fundamental aspects: 1) the visualization functionality included in the individual VATs for displaying the data and 2) the unified user based on multiple coordinated views [Röh+23] to accessing the workflow, tools, and parameters. Additional views further provided information about the active connections between tools, the intermediate results obtained, and the progress of the analysis session.

For the coherent arrangement of individual VAT views on the screen, the idea of

a system that allows access, modeling, and computation of layouts for content from multiple tools was followed [EST19]. Notably, this features automatic display layouts by optimizing various quality criteria. This leads to suitable view arrangement not only on a single screen but also in multi-screen environments. On top of that, we also took into account the existing layout preferences of users in specific application scenarios [Röh+23]. These include common tool layouts found in desktop environments, as well as custom layouts created by users for given tasks. Such pre-defined layouts turned out to be important in assisting users in their proven use during the application-centered evaluation from Section 6.3. Support for both automatic and pre-defined layouts, together with interactive layout modifications on demand, considerably reduced the effort required for manual tool view orchestration. The hypothesis that there are orientation rules for a unified UI, which can be defined without prior knowledge about the final implementation can thereby be seen as verified as this the clearly represented through the concept of user interface ensembles that were tested and applied in at least two of the three evaluation phases.

Overall, the concept of the presented unified user interface combines all this information via dedicated interactive view panels that can be dynamically added and freely arranged. It, therefore, enables the juxtaposing and linking of views between the different components, e.g., current work step, associated tools, and data exchange between them. In this way, all information is accessible to the user in a coherent manner, deeply referring to the "integrate what is necessary" on a visual level regarding the previously established coordination mantra.

7.5 Annotating the Data flow for Meaningful Information

In order to answer how to annotate the data flow to provide meaningful information, it is first important to define a general data annotation concept that can capture supplementary information about the data and the processes operating on the data. This was done following the findings from the visualization research group at the University of Rostock.

To this cause, Schmidt [SRS18] defined a morphological box to portray the interplay of annotation characteristics regarding different purposes and acquisition techniques for information processing and visualization. Together with the visualization experts, we discussed the integration of annotations into a knowledge generation model [Sac+17] and depicted obstacles that accompany the use of annotations, particularly concerning the visualization with different certainty levels. Based on this, Schmidt designed and implemented tailored annotation methods for data preprocessing, data cleansing, and data exploration [Sch+19a], which adhere to the principle of being as automatic as possible while enhancing trust in the data through reliability and transparency. Doing this, he described automatically generated annotations on data value redundancy information and discrepancy resolution [SRS20].

Following up on this, it was necessary to determine to which extent automated

annotations can potentially be used and at which point user-generated annotations are required. This required an expansion of the concept for user-generated annotations onto additional process information produced by the users, the individual VATs, or the coordination framework itself. In collaboration with the ophthalmological research group at the University of Rostock, we evaluated this approach based on an example of heterogeneous, contradictory, and incomplete optical coherence tomography (OCT) data within clinical institutions [SRS20; Sch+21]. In discussion with the domain experts, it became apparent that the need for automatic annotation of data itself did not have a high priority. Instead, there were multiple requirements to create visual representations for the reproducible documentation of outcomes based on manual user interaction, including comments, screenshots, and graphical summaries. Therefore, the results of these discussions were used in the application-centered evaluation Section 6.3 to enhance the existing unified user interface with support for ophthalmologists in performing their workflows. The hypothesis that the analysis process benefits from supervised annotation mechanics can thereby be verified, as we received positive feedback from multiple domain experts during this evaluation phase.

Together, we were able to not only define the transfer of raw data but also user comments, parameter choices, and process knowledge to other users, other tools, and the unified UI. With the support of the annotation-enriched tool, domain experts were able to understand the results and validate their findings from several thousand patients efficiently, thus avoiding generating possibly biased multi-center data. Even though this adaptation was necessary, the annotation concept behind the implemented platform should probably include further automation techniques in the future, for example, to utilize information from the metadata of data sources for the interactive analysis process.

7.6 Controlling User Interface Ensembles

In order to answer, how the user can interactively control the UI ensembles, this research particularly focused on increasing the effectiveness of steering multiple VATs through global control and user support.

Regarding the control of parameters, two fundamental options have been included in the existing implementations: 1) integrating global control panels and 2) utilizing existing control interfaces. For the first option, global UI components have been investigated to capture common parameters for all tools and allow changing their values via a separate central control panel. For the second option, the existing tool UIs have been used to control specific parameters and distribute changes across views. This was made possible by the adaptation of the ReVize library [HS19] and Vega-Lite descriptors [Sat+17] to represent and exchange all parameters used in the toolchain for workflow-centered evaluation in Section 6.2. We demonstrated the utility of both options in the Health@Hand software [Non+19] (e.g., setting attribute visibility across different data visualizations via a global control panel) and in the general data analysis use case [Non+22] (e.g., adjusting color scales and filtering subsets via the ColorBrewer [HB03] and Knime [Ber+08b] interfaces). The users were able to adjust general parameters using the global panels or

specific parameters via the tool UIs they were already familiar with, while the solution offered the benefit of automatically propagating the changes made. Both options have thus helped to replace repetitive operations in the individual control of multiple tools.

Regarding the user support, investigations in terms of smooth display transitions and annotations were conducted. Display transitions support the user beyond the arrangement of multiple tool views via smoothly animated transformations between visualizations when switching from one tool to the next. Together with other researchers from the University of Rostock, we conceptualized and formalized several aspects for smooth display transitions as a way to support users in dealing with multiple visualizations and launched a call for future research [TS20]. The annotation exchange from Section 7.5 helped users capture comments, parameter choices, work results, and histories in two ophthalmology use cases [Sch+19a; Röh+23]. Finally, concepts for user support through coordinated visual brushing and linking across multiple tools were explored in a student thesis [Zie19]. In this work, a central workflow controller connects two or more individual tools through event channels described in a linking file. The selection events generated by a tool are then processed by the controller and forwarded to one or more tools specified in the file. The receiving tools are updated to reflect the selection made, e.g., by visually highlighting the data parts in their output. Such visual cues facilitate comprehension when the same data is displayed in multiple outputs and thus assist user orientation when working with multiple tools in sequence over consecutive work steps or parallel within one step. The ideas and prototypical implementation of the bachelor thesis have been integrated and tested in the Unified UI of the Health@Hand framework [Non+19]. The hypothesis that there are options for interactively controlling local and global parameters during the analysis, which would benefit tool coordination, can thereby be seen as verified since it was used in commonly applied features during the evaluation. However, it is clear to see that there is high desire for more control parameter that are furthermore not pre-defined by the system itself but instead automatically identified and made available to assist the domain expert during their analysis.

Altogether, new strategies for controlling the display through global settings and visual cues have been explored throughout this work. It was particularly focused on increasing effectiveness through global control and user support. For this purpose, the new iteration of the AnyProc interface built on the results of data exchange channels (see Section 7.1) and flow-oriented annotation concepts (see Section 7.5). While the unified UI makes the developed solutions available to the user, the automated coordination and exchange mechanisms establish the necessary connections and ensure the exchange of information between tools. This provided the required mechanisms for exchanging control parameters alongside the data, as well as the first prototypical cross-tool annotation capabilities.

8 Conclusion

This chapter summarises the findings of the scientific concepts in this thesis and provides an outlook on future research in the field of coordinated visual analytics tools.

8.1 Summary

This thesis presented a holistic view on general-purpose coordination methods for the coupling of otherwise independent visual analytics tools in interactively controllable analytical toolchains.

The declared goal of this research was to establish research questions, the answers to which would allow for the re-use of existing visual analytics tools based on a domain user's workflow independent of the underlying individual implementation. This work, therefore, aimed to investigate how data inputs and outputs of VATs could be leveraged to establish data-level coordination mechanisms and how said outcomes could be merged via a coordination graph to define a unified user interface.

A brief overview of the related work in Chapter 2 has shown a wider variety of flow-oriented approaches for the coupling and coordination of data and views. Yet, the current literature still does not sufficiently address some combinations of aspects—especially the coordination of independent visual analytics tools without their integration in monolithic frameworks—and running different analysis tasks without manually switching between necessary visual analytics tools.

Based on these observations, different aspects of the separation of concern in the interactive visual analysis were identified and addressed:

- Conceptual separation of applied usage, data management, and functionality
- Spatial separation of stored data from multiple sources and displayed information from multiple views
- Temporal separation of defined tasks, performed actions, and received responses

This lead to the definition of a new mantra to "integrate what is necessary and couple what is possible". The idea behind it was followed at each step forward to stay independent of specific implementations.

This type of coupling among independent tools is declared as *lightweight coordination*, as it is minimally-invasive, pair-wise, and opportunistic in utilizing whichever interface a VAT offers. Following the concepts of nested visualization design [Mun15] and directness

in interactive visual data analysis [TS20], the lightweight nature is further enhanced by a layered attribution.

The previously mentioned research questions are analyzed following this flow-oriented structure and answered on two levels: 1) on the data level to facilitate the exchange of data between different tools, and 2) on the view level to arrange the visual output generated by different tools.

Based on the investigation and the resulting models, a suitable setup of the multiple tool user interfaces (UIs) was developed. This setup is meant to be seen as a unified interface for visual analytics since it supports the combination of VATs and allows access to data and functions of all tools to be used within a given analytical workflow. It, therefore, serves data-flow-oriented mechanisms for unified access to otherwise separately used individual VATs rather than developing a general-purpose application that permanently integrates standalone tools into a fully-fledged software solution. This concept enables users with limited or non-existent technical experience to integrate the functionalities of existing VATs into their workflow with minimal or, ideally, no implementation-related adjustments.

The proposed model was implemented as an open-source software solution in an extensible editor called the Analytical Process Configurator (AnyProc). The editor is thereby based on the concept of lightweight coordination and enables a loose coupling of multiple independent VATs in extensive analytical processes. The presented process model enables the abstraction and customization of modular coordination components to emphasize the analytical task while still obtaining the provenance over a graph structure. This facilitates the creation and configuration of analytical tasks based on the domain expert's needs and allows for the adjustment and re-configuration during analysis in order to gain new insights or change the desired outcomes of a visual analysis. All in all, it provides an easy-to-use interface that simplifies the temporal coupling of VATs similar to joint presentation software, while also remaining highly adaptable in terms of the number of imported VATs or data sources as well as to the size and complexity of created toolchains.

For evaluation purposes, the methodologies of this thesis have been demonstrated for three specific medical data analysis scenarios showcasing the proposed design process, coordination model, and UI setups. This was performed as a stage-wise evaluation that proved useful by testing developed coordination mechanisms and view layouts in isolation with a user group of technical experts, combining existing VATs in coordinated toolchains with technical and domain experts and presenting and comparing a functional prototype to manual toolchaining for the specific workflow with domain experts. To this cause, the presented coordination methods are assessed against the three medical use cases, each with a specific VAT set, a concrete domain workflow, and clear analysis objectives of said evaluation stage.

The evaluation (see Chapter 6) showed that the initial coordination mechanisms tested in isolation had to be founded by an underlying concept model. The required multifaceted model for the layered orchestration of independent VATs was well received by the visualization experts. Further, it proved useful to specific workflows of domain users, such as for the analysis of ophthalmology. The approach lifted the conceptual,

spatial, and temporal separation of visual analytics tasks and helped the users to perform their workflow without cumbersome interference.

In the end, the stage-wise evaluation process helped all involved parties by either solving analysis tasks more efficiently or finding hurdles for the practical usage of this approach that may need to be addressed in the future.

The results specifically emphasize that suitable coordination means are crucial for working with multiple tools in interactive data analysis workflows in different application domains. Through this, the defined hypothesis has been (partly) proven/falsified, and therefore, the research questions have been answered satisfactorily.

In conclusion, it can therefore safely stated that the goal of this research has been met and that it can be seen as a stepping stone for the further development of sustainable process chains in which independent visual analytics tools are blended together.

This led to new insights and prompted new ideas for further improvement regarding the concept of visual analytics toolchains. What remains is the application of the presented methods in various research domains outside the presented medical field to improve existing user workflows and other additional research regarding the automated handling and recommendation of information that have not been included in this research.

The solutions have been made publicly available as open-source projects on GitHub to provide access and encourage future use of this work. The repositories provided include a general description of the UnIVA project [Nonb], the AnyProc software for creating and executing toolchains [Nona], and the ReVize library for managing general data exchange [HS].

8.2 Future Work

It is out of question that further research regarding the combination and coordination of independent VATs should be considered. In fact, the evaluations at the last two stages have been intended to help specify interesting topics for future investigations.

8.2.1 Spread into Various Application Fields

While collaborating with various institutions, the primary evaluation feedback for the approaches was received from the medical application sector. Although this feedback has come from different disciplines of medical work, it can still be assumed that the domain is subject to a certain mindset. Certain application sectors, like manufacturing, might prioritize real-time data processing and predictive analytics, while other sectors, like environmental monitoring, may introduce new challenges related to handling large-scale, dynamic datasets. This could influence the scope of the toolchains as well as the type of data processing or the visual representation. Even with the prove that the methods presented in this work are independent of a specific domain, it is not clear how the users from different application sector would employ such methods in their workflow.

Looking forward, the approach of this work should be employed to model visual analysis tasks in various application sectors. Together with the feedback gathered

from various domain experts, one could define rule sets for different requirements when configuring VAT toolchains. This might lead to ways for simplifying toolchains, which in turn would increase efficiency while maintaining functionality. Additionally, such rule sets might help to identify the knowledge gap between the previously defined user roles from Subsection 3.3.2 in order to find mechanisms that would further reduce the involvement of the technical expert. Following this idea in the long term would most likely lead to more modular and interoperable tool ecosystems, capable of seamlessly integrating with the diverse workflows of the domain experts themselves.

8.2.2 Enhancements for the Data Connectivity and modular Data Preparation

In order to support the growing complexity and diversity of data used in modern visual analytics workflows, the coordination model should evolve to better handle both data connectivity and preparation. While the current implementation successfully integrates multiple tools by leveraging their existing interfaces, further refinement is needed to accommodate increasingly heterogeneous data types and sources. As workflows expand across disciplines and use cases, the challenge of preparing and connecting disparate data inputs will grow more pronounced.

In the future, this should be refined by enabling the integration of even more diverse data sources, which could include unstructured, semi-structured, and real-time data streams. The ability to seamlessly incorporate such varied data types would require improvements for AnyProcs data interoperability implementation. Moreover, enhancing the parameter exchange between tools becomes critical as the variety of data grows. A key approach here is the use of metadata and data descriptors, which can serve as a universal "language" for communicating the structure, type, and purpose of data between independent tools, allowing users to focus more on analysis rather than on setting up data connections.

One point that has not yet been considered in the context of this work is artificial intelligence methods for data preparation in toolchains. Smart technologies in this regard could for example not only pre-filter the data but also suggest it for re-use during the analysis. Going even further, it would be useful to also support automatic data mining initiatives in terms of modular methods for data cleaning, integration, data fusion etc. Nonetheless, these challenges are non-trivial and demand considerable development effort before modular data mining can be effectively incorporated into a visual analytics platform.

8.2.3 Delayed Data Transfer for Progressive Analytics

When working with large datasets and complex analyses, the analyst does not want to wait minutes or even hours if they are exploring the provided information. In such cases, a delayed data transfer approach becomes crucial, allowing for immediate interaction with initial results while more detailed data is processed in the background. Systems like

Hive [Thu+09] have previously explored trading off quality for speed, offering the ability to present initial results quickly, followed by more detailed data as processing continues.

Regarding analytical toolchains, there is significant potential in enhancing this concept through lightweight coordination models developed in this research. A key direction would be to allow analytical components to provide an initial, coarse overview of the data as soon as it becomes available before progressively refining the results as more data is transferred and processed. This staged approach would enable the analyst to interact with a preliminary visual representation, guiding their exploration without delay and refining it as necessary based on the evolving analysis.

Ideally, the scientist will be able to plug in almost any scientific data source and computational service into a workflow, inspect and visualize data on the fly, make parameter changes when necessary, and re-run only the affected components [AKD10]. This would not only improve the interactivity of large-scale analysis but also reduce computational overhead by focusing on areas of interest as they emerge during exploration.

In the future, more sophisticated data management layers within the coordination framework should be explored to support progressive data transfer and pre-processing for upcoming analysis steps in the toolchain. Through this, the presented editor would need to dynamically prioritize certain parts of the data based on user interactions and the stage of the workflow. For example, in a real-time analytics scenario, the system might initially present high-level trends or anomalies, allowing the user to drill down into finer details as needed without having to wait for the entire dataset to be processed upfront.

8.2.4 Intelligent Guidance for Annotations and Views

While the collaborative efforts have shaped a platform for the configuration and execution of analytics processes, it would be beneficial to further annotate supplementary information for the data and the processes operating on the data. Such annotations not only represent the thought processes of domain and technical experts but also illustrate the progression of the system behind these analyses. The current manual user-generated annotation concept is therefore only the first step in this regards as annotations could be placed on onto additional process information produced by the users, the individual VATs, or the coordination framework itself.

Existing methodologies, such as modular degree-of-interest specifications, have been effectively employed in the visual analysis of large dynamic networks [Abe+14]. These approaches could be adapted to prioritize VATs at different stages of toolchains so that domain users can semi-automatically suggest the right tools at the right time. By intelligently guiding users through the analysis process, the system can enhance the efficiency and effectiveness of data exploration.

This, coupled with strategies for highlighting critical information and reducing visual clutter, would facilitate the construction of more complex analytical processes. By streamlining the visual representation of data, users can concentrate on the most relevant aspects of their analysis, minimizing the distraction of extraneous details.

Future research should explore the potential extent of automated annotations, determining the points at which user-generated annotations are necessary. This investigation

could include examining saliency factors to derive data subsets based on the various coordination types clarified in Subsection 3.2.1. Such advancements would lead to a more intuitive and streamlined analytical experience, ultimately benefiting users as they navigate through their data-driven tasks.

Bibliography

[Abe+14] James Abello, Steffen Hadlak, Heidrun Schumann, and Hans-Jörg Schulz. "A Modular Degree-of-Interest Specification for the Visual Analysis of Large Dynamic Networks". In: *Transactions on Visualization and Computer Graphics* 20.3 (2014), pp. 337–350. ISSN: 1941-0506. DOI: 10.1109/tvcg.2013.109.

[ABS99] Serge Abiteboul, Peter Buneman, and Dan Suciu. *Data on the Web: from relations to semistructured data and XML*. Morgan Kaufmann, 1999. ISBN: 155860622x.

[Ade08] Herman J Ader. "Advising on Research Methods: A Consultant's Companion". In: vol. 14. Johannes van Kessel Publications, 2008. Chap. Phases and initial steps in data analysis, pp. 333–356.

[Aeh+13] Mario Aehnelt, Sebastian Bader, Gernot Ruscher, Frank Krüger, Bodo Urban, and Thomas Kirste. "Situation Aware Interaction with Multimodal Business Applications in Smart Environments". In: *Human Interface and the Management of Information. Information and Interaction for Learning, Culture, Collaboration and Business*. Ed. by David Hutchison, Takeo Kanade, and Josef Kittler. Springer, 2013, pp. 413–422. ISBN: 978-3-642-39226-9.

[AGL05] James Ahrens, Berk Geveci, and Charles Law. "Paraview: An end-user tool for large data visualization". In: *The visualization handbook* (2005). DOI: 10.1016/b978-012387582-2/50038-1.

[Ahr+01] James Ahrens, Kristi Brislawn, Ken Martin, Berk Geveci, Charles C. Law, and Michael E. Papka. "Large-scale data visualization using parallel data streaming". In: *Computer Graphics and Applications* 21.4 (2001), pp. 34–41. ISSN: 0272-1716. DOI: 10.1109/38.933522.

[Ahr+07] James P. Ahrens, Nehal Desai, Patrick S. McCormick, Ken Martin, and Jonathan Woodring. "A modular extensible visualization system architecture for culled prioritized data streaming". In: *Visualization and Data Analysis 2007*. Ed. by Robert F. Erbacher, Jonathan C. Roberts, Matti T. Gröhn, and Katy Börner. SPIE, 2007, p. 64950i. DOI: 10.1117/12.706325.

[AKD10] Anastasia Ailamaki, Verena Kantere, and Debabrata Dash. "Managing scientific data". In: *Communications of the ACM* 53.6 (2010), pp. 68–78.

[Ali+08] Kamran Ali, Knut Hartmann, Georg Fuchs, and Heidrun Schumann. "Adaptive Layout for Interactive Documents". In: *Smart Graphics*. Ed. by Andreas Butz, Brian Fisher, Antonio Krüger, Patrick Olivier, and Marc Christie. Vol. 5166. Lecture Notes in Computer Science. Springer, 2008, pp. 247–254. ISBN: 978-3-540-85410-4. DOI: 10.1007/978-3-540-85412-8_24.

[Ali+09] Muhammad Intizar Ali, Reinhard Pichler, Hong-Linh Truong, and Schahram Dustdar. "DeXIN: An extensible framework for distributed XQuery over heterogeneous data sources". In: *International Conference on Enterprise Information Systems*. Springer, 2009, pp. 172–183. DOI: 10.1007/978-3-642-01347-8_15.

[Ali+11] Muhammad Intizar Ali, Reinhard Pichler, Hong-Linh Truong, and Schahram Dustdar. "On Integrating Data Services Using Data Mashups". In: *Advances in Databases*. Ed. by Alvaro A. A. Fernandes, Alasdair J. G. Gray, and Khalid Belhajjame. Vol. 7051. Lecture Notes in Computer Science. Springer, 2011, pp. 132–135. ISBN: 978-3-642-24576-3. DOI: 10.1007/978-3-642-24577-0_14.

[Alo+97] Gustavo Alonso, Divyakant Agrawal, A. El Abbadi, and C. Mohan. "Functionality and Limitations of Current Workflow Management Systems". In: *Unpublished on Semantic Scholar* (1997). URL: https://api.semanticscholar.org/CorpusID:7089261.

[And+07] G. Andrienko, N. Andrienko, P. Jankowski, D. Keim, M.–J. Kraak, A. MacEachren, and S. Wrobel. "Geovisual analytics for spatial decision support: Setting the research agenda". In: *International Journal of Geographical Information Science* 21.8 (2007), pp. 839–857. ISSN: 1365-8816. DOI: 10.1080/13658810701349011.

[Ang+18] Marco Angelini, Giuseppe Santucci, Heidrun Schumann, and Hans-Jörg Schulz. "A Review and Characterization of Progressive Visual Analytics". In: *Informatics* 5.3 (2018), p. 31. DOI: 10.3390/informatics5030031.

[Are14] Marcelo Arenas. *Foundations of data exchange*. Cambridge University Press, 2014. ISBN: 9781139060158. DOI: 10.1017/cbo9781139060158.

[Arg+00] Danielle Argiro, Steve Kubica, Mark Young, and Steve Jorgensen. "Khoros: An Integrated Development Environment for Scientific Computing and Visualization". In: *Enabling Technologies for Computational Science*. Ed. by Elias N. Houstis, John R. Rice, Efstratios Gallopoulos, and Randall Bramley. Springer, 2000, pp. 147–157. ISBN: 978-1-4615-4541-5. DOI: 10.1007/978-1-4615-4541-5_12.

[ASS19] Mallika Agarwal, Arjun Srinivasan, and John Stasko. "VisWall: Visual Data Exploration Using Direct Combination on Large Touch Displays". In: *Visualization Conference (VIS)*. IEEE, 2019. ISBN: 978-1-7281-4941-7. DOI: 10.1109/visual.2019.8933673.

[ASU13] Mario Aehnelt, Hans-Jörg Schulz, and Bodo Urban. "Towards a Contextualized Visual Analysis of Heterogeneous Manufacturing Data". In: *Advances in Visual Computing*. Vol. 8034. Lecture Notes in Computer Science. Springer, 2013, pp. 76–85. ISBN: 978-3-642-41938-6. DOI: 10.1007/978-3-642-41939-3_8.

[Aus08] Matt Austin. "Graphics of Large Datasets: Visualizing a Million by A. Unwin, M. Theus, and H. Hofmann". In: *Biometrics* 64.1 (2008), pp. 315–316. ISSN: 0006-341x. DOI: 10.1111/j.1541-0420.2008.00962_8.x.

[Baï+19] Karim Baïna, Boualem Benatallah, Fabio Casati, and Farouk Toumani. "Model-Driven Web Service Development". In: *Active flow and combustion control 2018*. Ed. by Rudibert King. Vol. 141. Notes on Numerical Fluid Mechanics and Multidisciplinary Design. Springer, 2019, pp. 290–306. ISBN: 978-3-319-98176-5. DOI: 10.1007/978-3-540-25975-6_22.

[Bar+07] James Christopher Bare, Paul T. Shannon, Amy K. Schmid, and Nitin S. Baliga. "The Firegoose: two-way integration of diverse data from different bioinformatics web resources with desktop applications". In: *BMC bioinformatics* 8 (2007), p. 456. DOI: 10.1186/1471-2105-8-456.

[Bar+13] Mihai Barbulescu, Ramona-Oana Grigoriu, Ionela Halcu, Giorzian Neculoiu, Virginia Cristiana Sandulescu, Mariana Marinescu, and Viorel Marinescu. "Integrating of structured, semi-structured and unstructured data in natural and build environmental engineering". In: *RoEduNet International Conference 11th Edition: Networking in Education and Research*. IEEE, 2013, pp. 1–4. ISBN: 978-1-4673-6116-3. DOI: 10.1109/RoEduNet.2013.6511738.

[Bat+20] Steven Batt, Tara Grealis, Oskar Harmon, and Paul Tomolonis. "Learning Tableau: A data visualization tool". In: *The Journal of Economic Education* 51.3-4 (2020), pp. 317–328. ISSN: 0022-0485. DOI: 10.1080/00220485.2020.1804503.

[Bau+16] Peter Baumann, Paolo Mazzetti, Joachim Ungar, Roberto Barbera, Damiano Barboni, Alan Beccati, Lorenzo Bigagli, Enrico Boldrini, Riccardo Bruno, Antonio Calanducci, Piero Campalani, Oliver Clements, Alex Dumitru, Mike Grant, Pasquale Herzig, George Kakaletris, John Laxton, Panagiota Koltsida, Kinga Lipskoch, Alireza Rezaei Mahdiraji, Simone Mantovani, Vlad Merticariu, Antonio Messina, Dimitar Misev, Stefano Natali, Stefano Nativi, Jelmer Oosthoek, Marco Pappalardo, James Passmore, Angelo Pio Rossi, Francesco Rundo, Marcus Sen, Vittorio Sorbera, Don Sullivan, Mario Torrisi, Leonardo Trovato, Maria Grazia Veratelli, and Sebastian Wagner. "Big Data Analytics for Earth Sciences: the EarthServer approach". In: *International Journal of Digital Earth* 9.1 (2016), pp. 3–29. DOI: 10.1080/17538947.2014.1003106.

[BB12] James Christopher Bare and Nitin S. Baliga. "Architecture for interoperable software in biology". In: *Briefings in Bioinformatics* 15.4 (2012), pp. 626–636. DOI: 10.1093/bib/bbs074.

[BDS05] Boualem Benatallah, Marlon Dumas, and Quan Z. Sheng. "Facilitating the Rapid Development and Scalable Orchestration of Composite Web Services". In: *Distributed and Parallel Databases* 17.1 (2005), pp. 5–37. ISSN: 0926-8782. DOI: 10.1023/b:dapd.0000045366.15607.67.

[Beh+19] Michael Behrisch, Dirk Streeb, Florian Stoffel, Daniel Seebacher, Brian Matejek, Stefan Hagen Weber, Sebastian Mittelstadt, Hanspeter Pfister, and Daniel Keim. "Commercial Visual Analytics Systems-Advances in the Big Data Analytics Field". In: *Transactions on Visualization and Computer Graphics* 25.10 (2019), pp. 3011–3031. DOI: 10.1109/tvcg.2018.2859973.

[Beh21] Giulio Behringer. "Gruppierung der Analyseergebnisse für datenflussorientierte Werkzeugketten". Master Thesis. Hochschule Heilbronn, 2021.

[Bel+08] Khalid Belhajjame, Katy Wolstencroft, Oscar Corcho, Tom Oinn, Franck Tanoh, Alan William, and Carole Goble. "Metadata Management in the Taverna Workflow System". In: *Eighth International Symposium on Cluster Computing and the Grid (CCGRID)*. IEEE, 2008, pp. 651–656. DOI: 10.1109/ccgrid.2008.17.

[Ben+09] Werner Benger, Georg Ritter, Marcel Ritter, and Wolfram Schoor. "Beyond the Visualization Pipeline". In: *5th High-End Visualization Workshop*. Lehmanns Media-LOB.de, 2009. ISBN: 978-3-86541-330-7.

[Ben+11] Veronique Benzaken, Jean-Daniel Fekete, Pierre-Luc Hemery, Wael Khemiri, and Ioana Manolescu. "EdiFlow: Data-intensive interactive workflows for visual analytics". In: *27th International Conference on Data Engineering (ICDE 2011)*. IEEE, 2011, pp. 780–791. ISBN: 978-1-4244-8959-6. DOI: 10.1109/icde.2011.5767914.

[Ber+08a] Michael R. Berthold, Nicolas Cebron, Fabian Dill, Thomas R. Gabriel, Tobias Kötter, Thorsten Meinl, Peter Ohl, Christoph Sieb, Kilian Thiel, and Bernd Wiswedel. "KNIME Version 2.0 and Beyond". In: *Data Analysis, Machine Learning and Applications*. Ed. by Christine Preisach, Hans Burkhardt, Lars Schmidt-Thieme, and Reinhold Decker. Studies in Classification, Data Analysis, and Knowledge Organization. Springer, 2008, pp. 319–326. ISBN: 978-3-540-78239-1. DOI: 10.1007/978-3-540-78246-9_38.

[Ber+08b] Michael R. Berthold, Nicolas Cebron, Fabian Dill, Thomas R. Gabriel, Tobias Kötter, Thorsten Meinl, Peter Ohl, Christoph Sieb, Kilian Thiel, and Bernd Wiswedel. "KNIME: The Konstanz Information Miner". In: *Data Analysis, Machine Learning and Applications*. Studies in Classification, Data Analysis, and Knowledge Organization. Springer, 2008, pp. 319–326. ISBN: 978-3-540-78239-1. DOI: 10.1007/978-3-540-78246-9_38.

[BHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. "The semantic web". In: *Scientific american* 284.5 (2001), pp. 34–43.

[BL10] Enrico Bertini and Denis Lalanne. "Investigating and reflecting on the integration of automatic data analysis and visualization in knowledge discovery". In: *SIGKDD Explorations Newsletter* 11.2 (2010), pp. 9–18. ISSN: 1931-0145. DOI: 10.1145/1809400.1809404.

[Bla+12] Erik P. Blasch, Dale A. Lambert, Pierre Valin, Mieczyslaw M. Kokar, James Llinas, Subrata Das, Chee Chong, and Elisa Shahbazian. "High Level Information Fusion (HLIF): Survey of models, issues, and grand challenges". In: *Aerospace and Electronic Systems Magazine* 27.9 (2012), pp. 4–20. ISSN: 0885-8985. DOI: 10.1109/maes.2012.6366088.

[BM02] Vladimir Batagelj and Andrej Mrvar. "Pajek - Analysis and Visualization of Large Networks". In: *Graph Drawing*. Ed. by Petra Mutzel, Michael Jünger, and Sebastian Leipert. Springer, 2002, pp. 477–478. ISBN: 978-3-540-45848-7.

[BP08] Charl P. Botha and Frits H. Post. "Hybrid scheduling in the DeVIDE dataflow visualisation environment". In: *Proceedings of Simulation and Visualization*. SCS Publishing House Erlangen, 2008.

[Bra+08] Tim Bray, Jean Paoli, C. Michael Sperberg-McQueen, Eve Maler, and Franois Yergeau. *Extensible Markup Language (XML) 1.0*. https://www.w3.org/TR/xml. Last accessed 06-11-2024. 2008.

[Bre+22] Matthew Brehmer, Bongshin Lee, John Stasko, and Christian Tominski. "Interacting with Visualization on Mobile Devices". In: *Mobile Data Visualization*. Ed. by Bongshin Lee, Raimund Dachselt, Petra Isenberg, and Eun Kyoung Choe. AK Peters Visualization Ser. CRC Press LLC, 2022, pp. 67–110. ISBN: 9781003090823. DOI: 10.1201/9781003090823-3.

[Bru+13] Josep Maria Brunetti, Sören Auer, Roberto García, Jakub Klímek, and Martin Nečaský. "Formal Linked Data Visualization Model". In: *Proceedings of International Conference on Information Integration and Web-based Applications & Services*. Ed. by Edger Weippl, Maria Indrawan-Santiago, Matthias Steinbauer, Gabriele Kotsis, and Ismail Khalil. ACM, 2013, pp. 309–318. ISBN: 9781450321136. DOI: 10.1145/2539150.2539162.

[BSD14] Athman Bouguettaya, Quan Z. Sheng, and Florian Daniel, eds. *Web Services Foundations*. Springer, 2014. ISBN: 9781461475170. DOI: 10.1007/978-1-4614-7518-7.

[But+08] Thomas Butkiewicz, Remco Chang, Zachary Wartell, and William Ribarsky. "Visual Analysis and Semantic Exploration of Urban LIDAR Change Detection". In: *Computer Graphics Forum* 27.3 (2008), pp. 903–910. ISSN: 01677055. DOI: 10.1111/j.1467-8659.2008.01223.x.

[C U+89] C. Upson, T. A. Faulhaber, D. Kamins, D. Laidlaw, D. Schlegel, J. Vroom, R. Gurwitz, and A. van Dam. "The application visualization system: a computational environment for scientific visualization". In: *Computer Graphics and Applications* 9.4 (1989), pp. 30–42. ISSN: 1558-1756. DOI: 10.1109/38.31462.

[Cai+03] Deng Cai, Shipeng Yu, Ji-Rong Wen, and Wei-Ying Ma. "Extracting Content Structure for Web Pages Based on Visual Representation". In: *Web Technologies and Applications*. Ed. by Xiaofang Zhou, Maria E. Orlowska, and Yanchun Zhang. Vol. 2642. Lecture Notes in Computer Science. Springer, 2003, pp. 406–417. ISBN: 978-3-540-02354-8. DOI: 10.1007/3-540-36901-5_42.

[Cal+08] Steven P. Callahan, Juliana Freire, Carlos E. Scheidegger, Cláudio T. Silva, and Huy T. Vo. "Towards Provenance-Enabling ParaView". In: *Provenance and annotation of data and processes*. Ed. by Juliana Freire, David Koop, and Luc Moreau. Lecture Notes in Computer Science. Springer, 2008, pp. 120–127. ISBN: 978-3-540-89965-5.

[Cal+09] Diego Calvanese, Giuseppe de Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. "Conceptual Modeling for Data Integration". In: *Conceptual modeling: foundations and applications*. Ed. by Alexander T. Borgida, Vinay K. Chaudhri, Paolo Giorgini, and Eric S. Yu. Vol. 5600. Springer, 2009, pp. 173–197. ISBN: 978-3-642-02462-7. DOI: 10.1007/978-3-642-02463-4_11.

[Cal02] Calì, Andrea and Calvanese, Diego and De Giacomo, Giuseppe and Lenzerini, Maurizio. "Data Integration under Integrity Constraints". In: *Advanced Information Systems Engineering*. Ed. by Anne Banks Pidduck, M. Tamer Ozsu, John Mylopoulos, and Carson C Woo. Elsevier, 2002, pp. 262–279. DOI: 10.1016/S0306-4379(03)00050-4.

[CC07] Christopher Collins and Sheelagh Carpendale. "VisLink: revealing relationships amongst visualizations". In: *Transactions on Visualization and Computer Graphics* 13.6 (2007), pp. 1192–1199. ISSN: 1077-2626. DOI: 10.1109/tvcg.2007.70521.

[CD97] Surajit Chaudhuri and Umeshwar Dayal. "An overview of data warehousing and OLAP technology". In: *SIGMOD Record* 26.1 (1997), pp. 65–74. ISSN: 01635808. DOI: 10.1145/248603.248616.

[CFC17] R. Jordan Crouser, Lyndsey Franklin, and Kris Cook. "Rethinking Visual Analytics for Streaming Data Applications". In: *Internet Computing* 21.4 (2017), pp. 72–76. ISSN: 1089-7801. DOI: 10.1109/mic.2017.2911428.

[CG98] Peter Chamoni and Peter Gluchowski. "On-Line Analytical Processing (OLAP)". In: *Das Data Warehouse-Konzept*. Ed. by Harry Mucksch. Gabler, 1998, pp. 401–444. ISBN: 978-3-409-32216-4. DOI: 10.1007/978-3-322-99372-4{_}13.

[Cha+21] Joseph Chee Chang, Yongsung Kim, Victor Miller, Michael Xieyang Liu, Brad A. Myers, and Aniket Kittur. "Tabs.do: Task-Centric Browser Tab Management". In: *The 34th Annual Symposium on User Interface Software and Technology*. Ed. by Jeffrey Nichols. ACM Digital Library. ACM, 2021, pp. 663–676. ISBN: 9781450386357. DOI: 10.1145/3472749.3474777.

[Che06] Chaomei Chen. *Information Visualization: Beyond the Horizon*. Second Edition. SpringerLink Bücher. Springer London, 2006. ISBN: 9781846285790. DOI: 10.1007/1-84628-579-8.

[Chi+11] Hank Childs, Eric Brugger, Brad Whitlock, Jeremy Meredith, Sean Ahern, Kathleen Bonnell, Mark Miller, Gunther Weber, Cyrus Harrison, David Pugmire, Thomas Fogal, Christoph Garth, A. Sanderson, E. Wes Bethel, M. Durant, David Camp, Jean Favre, O. Rübel, Paul Navratil, and F. Vivodtzev. "VisIt: An End-User Tool for Visualizing and Analyzing Very Large Data". In: *Proceed SciDAC* (2011). Last Accessed: 06-11-2024, pp. 1–16.

[Chu+15] Haeyong Chung, Chris North, Sarang Joshi, and Jian Chen. "Four considerations for supporting visual analysis in display ecologies". In: *Conference on Visual Analytics Science and Technology (VAST)*. IEEE, 2015, pp. 33–40. DOI: 10.1109/vast.2015.7347628.

[Cig] Paolo Cignoni. *Meshlab*. http://meshlab.sourceforge.net. Last accessed 06-11-2024.

[CL10] Joseph A. Cottam and Andrew Lumsdaine. "Automatic Application of the Data-State Model in Data-Flow Contexts". In: *14th International Conference Information Visualisation (IV)*. Ed. by Ebad Banissi. IEEE, 2010, pp. 5–10. ISBN: 978-1-4244-7846-0. DOI: 10.1109/iv.2010.10.

[CMS99] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman, eds. *Readings in information visualization: Using vision to think*. ACM, 1999. ISBN: 978-1-55860-533-6.

[CR98] Ed Huai-Hsin Chi and J. T. Riedl. "An operator interaction framework for visualization systems". In: *Symposium on Information Visualization (Cat. No.98TB100258)*. IEEE, 1998, pp. 63–70. ISBN: 0-8186-9093-3. DOI: 10.1109/infvis.1998.729560.

[CRC07] Alan Cooper, Robert Reimann, and David Cronin. *About face 3: the essentials of interaction design*. John Wiley & Sons, 2007. ISBN: 9780470084113.

[CSH03] Christian Chabot, Chris Stolte, and Pat Hanrahan. "Tableau software". In: *Tableau software* 6 (2003). Last accessed 06-11-2024.

[CT05] Kristin A. Cook and James J. Thomas. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. 54th ed. Visual Representations and Interactions Technologies. IEEE, 2005. ISBN: 978-0769523231.

[Ćur+02] V. Ćurčin, Moustafa Ghanem, Yike Guo, Martin Köhler, Anthony Rowe, Jameel Syed, and Patrick Wendel. "Discovery net: towards a grid of knowledge discovery". In: *Proceedings of the eighth SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2002, pp. 658–663. DOI: 10.1145/775047.775145.

[CYM11] Tsung-Hsiang Chang, Tom Yeh, and Rob Miller. "Associating the visual representation of user interfaces with their internal structures and metadata". In: *Proceedings of the 24th annual symposium on User interface software and technology*. Ed. by Jeff Pierce. ACM, 2011, p. 245. ISBN: 9781450307161. DOI: 10.1145/2047196.2047228.

[Dar+16] Cinzia Daraio, Maurizio Lenzerini, Claudio Leporelli, Henk F. Moed, Paolo Naggar, Andrea Bonaccorsi, and Alessandro Bartolucci. "Data integration for research and innovation policy: an Ontology-Based Data Management approach". In: *Scientometrics* 106.2 (2016), pp. 857–871. ISSN: 0138-9130. DOI: 10.1007/s11192-015-1814-0.

[dB04] Selan dos Santos and Ken Brodlie. "Gaining understanding of multivariate and multidimensional data through visualization". In: *Computers & Graphics* 28.3 (2004), pp. 311–325. ISSN: 00978493. DOI: 10.1016/j.cag.2004.03.013.

[DBR11] Mark S. Doderer, Cory Burkhardt, and Kay A. Robbins. "SIDECACHE: Information access, management and dissemination framework for web services". In: *BMC research notes* 4 (2011), p. 182. DOI: 10.1186/1756-0500-4-182.

[Del+11] N. Del Rio, P. Da Pinheiro Silva, G. G. Leptoukh, and C. Lynnes. "Towards Infusing Giovanni with a Semantic and Provenance Aware Visualization System". In: *AGU Fall Meeting Abstracts* 2011 (2011). Last Accessed: 06-11-2024.

[DFW12] Morgan Dixon, James Fogarty, and Jacob Wobbrock. "A general-purpose target-aware pointing enhancement using pixel-level analysis of graphical interfaces". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Ed. by Joseph A. Konstan, Ed H. Chi, and Kristina Höök. ACM, 2012, p. 3167. ISBN: 978-1-4503-1015-4. DOI: 10.1145/2207676.2208734.

[DGR05] A. Das, J. Gehrke, and M. Riedewald. "Semantic approximation of data stream joins". In: *Transactions on Knowledge and Data Engineering* 17.1 (2005), pp. 44–59. ISSN: 1041-4347. DOI: 10.1109/tkde.2005.17.

[DHI12] AnHai Doan, Alon Halevy, and Zachary G. Ives. *Principles of data integration*. Morgan Kaufmann, 2012. ISBN: 9780124160446.

[Di +09] Giusy Di Lorenzo, Hakim Hacid, Hye-young Paik, and Boualem Benatallah. "Data integration in mashups". In: *ACM SIGMOD Record* 38.1 (2009), p. 59. ISSN: 01635808. DOI: 10.1145/1558334.1558343.

[DLF11] Morgan Dixon, Daniel Leventhal, and James Fogarty. "Content and hierarchy in pixel-based methods for reverse engineering interface structure". In: *Conference proceedings and extended abstracts / the 29th Annual CHI Conference on Human Factors in Computing Systems*. Ed. by Desney Tan, Geraldine Fitzpatrick, Carl Gutwin, Bo Begole, and Wendy A. Kellogg. ACM, 2011, p. 969. ISBN: 978-1-4503-0228-9. DOI: 10.1145/1978942.1979086.

[Dör+08] Marian Dörk, Sheelagh Carpendale, Christopher Collins, and Carey Williamson. "VisGets: Coordinated Visualizations for Web-based Information Exploration and Discovery". In: *Transactions on Visualization and Computer Graphics* 14.6 (2008), pp. 1205–1212. ISSN: 1941-0506. DOI: 10.1109/tvcg.2008.175.

[Dun+12] Cody Dunne, Nathalie Henry Riche, Bongshin Lee, Ronald Metoyer, and George Robertson. "GraphTrail: analyzing large multivariate, heterogeneous networks while supporting exploration history". In: *CHI Proceedings*. Ed. by Joseph A. Konstan, Ed H. Chi, and Kristina Höök. ACM, 2012, p. 1663. ISBN: 9781450310154. DOI: 10.1145/2207676.2208293.

[Dut16] Sudipa DuttaRoy. "SAP Business Analytics Suite of Products". In: *SAP Business Analytics*. Ed. by Sudipa Dutta Roy and Sudipa DuttaRoy. Apress, 2016, pp. 7–12. ISBN: 978-1-4842-1384-1. DOI: 10.1007/978-1-4842-1383-4_2.

[DYR10] Mark S. Doderer, Kihoon Yoon, and Kay A. Robbins. "SIDEKICK: Genomic data driven analysis and decision-making framework". In: *BMC bioinformatics* 11 (2010), p. 611. DOI: 10.1186/1471-2105-11-611.

[EB11] Micheline Elias and Anastasia Bezerianos. "Exploration Views: Understanding Dashboard Creation and Customization for Visualization Novices". In: *Human-computer interaction - INTERACT 2011*. Ed. by Pedro Campos, Nicholas Graham, Joaquim Jorge, Nuno Nunes, Philippe Palanque, and Marco Winckler. Vol. 6949. Lecture Notes in Computer Science. Springer, 2011, pp. 274–291. ISBN: 978-3-642-23767-6. DOI: 10.1007/978-3-642-23768-3{_}23.

[ED07] Geoffrey Ellis and Alan Dix. "A taxonomy of clutter reduction for information visualisation". In: *Transactions on Visualization and Computer Graphics* 13.6 (2007), pp. 1216–1223. ISSN: 1077-2626. DOI: 10.1109/tvcg.2007.70535.

[Eic+15] Christian Eichner, Thomas Nocke, Hans-Jörg Schulz, and Heidrun Schumann. "Interactive Presentation of Geo-Spatial Climate Data in Multi-Display Environments". In: *ISPRS International Journal of Geo-Information* 4.2 (2015), pp. 493–514. DOI: 10.3390/ijgi4020493.

[Eic00] Stephen G. Eick. "Visualizing multi-dimensional data". In: *SIGGRAPH Computer Graphics* 34.1 (2000), pp. 61–67. ISSN: 0097-8930. DOI: 10.1145/563788.604454.

[EK04] Thomas Ertl and Daniel Keim. "Wissenschaftliche Visualisierung – Ausgewählte Forschungsprojekte (Scientific Visualization – Selected Research Projects)". In: *it - Information Technology* 46.3 (2004), pp. 148–153. ISSN: 1611-2776. DOI: 10.1524/itit.46.3.148.34222.

[Ell+02] John Ellson, Emden Gansner, Lefteris Koutsofios, Stephen C. North, and Gordon Woodhull. "Graphviz— Open Source Graph Drawing Tools". In: *Graph Drawing*. Ed. by Petra Mutzel, Michael Jünger, and Sebastian Leipert. Springer, 2002, pp. 483–484. ISBN: 978-3-540-45848-7. DOI: 10.1007/3-540-45848-4_57.

[EN16] Ramez Elmasri and Sham Navathe. *Fundamentals of database systems*. Seventh edition. Pearson, 2016. ISBN: 978-1-292-09761-9. DOI: Ramez.

[Eng+06] Klaus Engel, Markus Hadwiger, Joe M. Kniss, and Christof Rezk-Salama. *Real-Time Volume Graphics*. 2006. DOI: 10.2312/egt.20061064.

[ENS15] Christian Eichner, Martin Nyolt, and Heidrun Schumann. "A Novel Infrastructure for Supporting Display Ecologies". In: *Advances in visual computing*. Ed. by George Bebis, Richard Boyle, Bahram Parvin, Darko Koracin, Ioannis Pavlidis, Rogerio Feris, Tim McGraw, Mark Elendt, Regis Kopper, Eric Ragan, Zhao Ye, and Gunther H. Weber. Lecture notes in computer science Image processing, computer vision, pattern recognition, and graphics. Springer, 2015, pp. 722–732. ISBN: 978-3-319-27863-6.

[ES00] Martin Ester and Jörg Sander. *Knowledge Discovery in Databases: Techniken und Anwendungen*. Springer eBook Collection Computer Science and Engineering. Springer, 2000. ISBN: 9783642583315. DOI: 10.1007/978-3-642-58331-5.

[EST19] Christian Eichner, Heidrun Schumann, and Christian Tominski. "Multi-display Visual Analysis: Model, Interface, and Layout Computation". In: *CoRR* abs/1912.08558 (2019). DOI: 10.48550/arXiv.1912.08558.

[EWS18] Maha El Meseery, Yuyao Wu, and Wolfgang Stuerzlinger. "Multiple Workspaces in Visual Analytics". In: *International Symposium on Big Data Visual and Immersive Analytics (BDVA)*. IEEE, 2018, pp. 1–12. ISBN: 978-1-5386-9194-6. DOI: 10.1109/bdva.2018.8534019.

[Fek+11] Jean-Daniel Fekete, Pierre-Luc Hemery, Thomas Baudel, and Jo Wood. "Obvious: A meta-toolkit to encapsulate information visualization toolkits — One toolkit to bind them all". In: *Conference on Visual Analytics Science and Technology (VAST)*. Ed. by Silvia Miksch. IEEE, 2011, pp. 91–100. ISBN: 978-1-4673-0014-8. DOI: 10.1109/vast.2011.6102446.

[Fek04] J.-D. Fekete. "The InfoVis Toolkit". In: *InfoVis*. Ed. by Matt Ward and Tamara Munzner. IEEE, 2004, pp. 167–174. DOI: 10.1109/infvis.2004.64.

[Fek13] Jean-Daniel Fekete. "Visual Analytics Infrastructures: From Data Management to Exploration". In: *Computer* 46.7 (2013), pp. 22–29. ISSN: 1558-0814. DOI: 10.1109/mc.2013.120.

[Fer+11] José Ignacio Fernández Villamor, Jacobo Blasco Garcia, Carlos Angel Iglesias Fernandez, and Mercedes Garijo Ayestaran. "A semantic scraping model for web resources-Applying linked data to web page screen scraping". In: *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence* 2 (2011), pp. 451–456. DOI: 10.5220/0003185704510456.

[Few06] Stephen Few. *Information dashboard design: The effective visual communication of data*. O'Reilly Media, Inc., 2006. ISBN: 978-0-596-10016-2.

[Fis+10] Danyel Fisher, Steven M. Drucker, Roland Fernandez, and Scott Ruble. "Visualizations everywhere: A multiplatform infrastructure for linked visualizations". In: *Transactions on Visualization and Computer Graphics* 16.6 (2010), pp. 1157–1163. DOI: 10.1109/tvcg.2010.222.

[Fou+14] Adam Fourney, Ben Lafreniere, Parmit Chilana, and Michael Terry. "InterTwine: Creating interapplication information scent to support coordinated use of software". In: *Proceedings of the 27th annual symposium on User interface software and technology - UIST '14*. Ed. by Hrvoje Benko, Mira Dontcheva, and Daniel Wigdor. ACM, 2014, pp. 429–438. ISBN: 9781450330695. DOI: 10.1145/2642918.2647420.

[Fou95] David Foulser. "IRIS Explorer". In: *SIGGRAPH Computer Graphics* 29.2 (1995), pp. 13–16. ISSN: 0097-8930. DOI: 10.1145/204362.204365.

[Fre+14] Steffen Frey, Filip Sadlo, Kwan-Liu Ma, and Thomas Ertl. "Interactive Progressive Visualization with Space-Time Error Control". In: *Transactions on Visualization and Computer Graphics* 20.12 (2014), pp. 2397–2406. ISSN: 1941-0506. DOI: 10.1109/tvcg.2014.2346319.

[FSL05] C. Forsell, S. Seipel, and M. Lind. "Simple 3D glyphs for spatial multivariate". In: *InfoVis 05*. Ed. by John Stasko and Matt Ward. IEEE, 2005, pp. 119–124. ISBN: 0-7803-9464-x. DOI: 10.1109/infvis.2005.1532137.

[Fuj+11] Pauline A. Fujita, Brooke Rhead, Ann S. Zweig, Angie S. Hinrichs, Donna Karolchik, Melissa S. Cline, Mary Goldman, Galt P. Barber, Hiram Clawson, Antonio Coelho, Mark Diekhans, Timothy R. Dreszer, Belinda M. Giardine, Rachel A. Harte, Jennifer Hillman-Jackson, Fan Hsu, Vanessa Kirkup, Robert M. Kuhn, Katrina Learned, Chin H. Li, Laurence R. Meyer, Andy Pohl, Brian J. Raney, Kate R. Rosenbloom, Kayla E. Smith, David Haussler, and W. James Kent. "The UCSC Genome Browser database: update 2011". In: *Nucleic acids research* 39 (2011). DOI: 10.1093/nar/gkq963.

[Gah+02] Mark Gahegan, Masahiro Takatsuka, Mike Wheeler, and Frank Hardisty. "Introducing GeoVISTA Studio: an integrated suite of visualization and computational methods for exploration and knowledge construction in geography". In: *Computers, Environment and Urban Systems* 26.4 (2002), pp. 267–292. ISSN: 0198-9715. DOI: 10.1016/s0198-9715(01)00046-1.

[GC09] Max Goebel and Michal Ceresna. "Wrapper Induction". In: *Encyclopedia of Database Systems*. Ed. by Ling Liu and M. Tamer Özsu. Springer, 2009, pp. 3560–3565. DOI: 10.1007/978-0-387-39940-9_1160.

[GKW14] Michael Glueck, Azam Khan, and Daniel J. Wigdor. "Dive in!" In: *14th Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Ed. by Matt Jones, Philippe Palanque, Albrecht Schmidt, and Tovi Grossman. ACM, 2014, pp. 561–570. ISBN: 978-1-4503-2473-1. DOI: 10.1145/2556288.2557195.

[Góm+13] John Gómez, Leyla J. García, Gustavo A. Salazar, Jose Villaveces, Swanand Gore, Alexander García, Maria J. Martín, Guillaume Launay, Rafael Alcántara, Noemi del-Toro, Marine Dumousseau, Sandra Orchard, Sameer Velankar, Henning Hermjakob, Chenggong Zong, Peipei Ping, Manuel Corpas, and Rafael C. Jiménez. "BioJS: an open source JavaScript framework for biological data visualization". In: *Bioinformatics (Oxford, England)* 29.8 (2013), pp. 1103–1104. DOI: 10.1093/bioinformatics/btt100.

[Got+08] David Gotz, Zhen Wen, Jie Lu, Peter Kissa, Michelle X. Zhou, Nan Cao, Wei Hong Qian, and Shi Xia Liu. "Interactive Poster: HARVEST-Visualization and Analysis for the Masses". In: *InfoVis Posters* (2008). Last Accessed: 06-11-2024.

[Göt+18] Aline Götze, Sophie von Keyserlingk, Sabine Peschel, Ulrike Jacoby, Corinna Schreiver, Bernd Köhler, Stephan Allgeier, Karsten Winter, Martin Röhlig, Anselm Jünemann, Rainer Guthoff, Oliver Stachs, and Dagmar-Christiane Fischer. "The corneal subbasal nerve plexus and thickness of the retinal layers in pediatric type 1 diabetes and matched controls". In: *Sci Rep* 8.1 (2018). DOI: 10.1038/s41598-017-18284-z.

[Gsc+14] Theresia Gschwandtner, Wolfgang Aigner, Silvia Miksch, Johannes Gärtner, Simone Kriglstein, Margit Pohl, and Nik 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*. 2014, p. 18. DOI: 10.1145/2637748.2638423.

[Gür+16] Didem Gürdür, Fredrik Asplund, Jad El-khoury, Frederic Loiret, and Martin Törngren. "Visual Analytics Towards Tool Interoperabilty - A Position Paper". In: *Visigrapp 2016*. Ed. by Nadia Magnenat-Thalmann, Paul Richard, Lars Linsen, Alexandru Telea, Sebastiano Battiato, Francisco Imai, and José Braz. Scitepress, 2016, pp. 139–145. ISBN: 978-989-758-175-5. DOI: 10.5220/0005751401390145.

[GW08] David Gotz and Zhen Wen. "Behavior-driven visualization recommendation". In: *Iui 2009*. Ed. by Daniel S. Weld. ACM Press, 2008, p. 315. ISBN: 9781605581682. DOI: 10.1145/1502650.1502695.

[GZ08] D. Gotz and M. X. Zhou. "Characterizing users' visual analytic activity for insight provenance". In: *Symposium on Visual Analytics Science and Technology*. IEEE, 2008, pp. 123–130. DOI: 10.1109/vast.2008.4677365.

[HA06] Jeffrey Heer and Maneesh Agrawala. "Software design patterns for information visualization". In: *Transactions on Visualization and Computer Graphics* 12.5 (2006), pp. 853–860. DOI: 10.1109/tvcg.2006.178.

[HA08] Jeffrey Heer and Maneesh Agrawala. "Design Considerations for Collaborative Visual Analytics". In: *Information Visualization* 7.1 (2008), pp. 49–62. ISSN: 1473-8716. DOI: 10.1057/palgrave.ivs.9500167.

[Hae88] Paul E. Haeberli. "ConMan: A visual programming language for interactive graphics". In: *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*. ACM, 1988, pp. 103–111. DOI: 10.1145/54852.378494.

[Hal01] Alon Y. Halevy. "Answering queries using views: A survey". In: *The VLDB Journal* 10.4 (2001), pp. 270–294. DOI: 10.1007/s007780100054.

[Han+13] Yanbo Han, Guiling Wang, Guang Ji, and Peng Zhang. "Situational data integration with data services and nested table". In: *Service Oriented Computing and Applications* 7.2 (2013), pp. 129–150. ISSN: 1863-2386. DOI: 10.1007/s11761-012-0103-5.

[Han06] Pat Hanrahan. "VizQL". In: *Proceedings of the SIGMOD international conference on Management of data*. Ed. by Clement Yu. ACM Conferences. ACM, 2006, p. 721. ISBN: 978-1-59593-434-5. DOI: 10.1145/1142473.1142560.

[HB03] Mark Harrower and Cynthia A. Brewer. "ColorBrewer.org: An online tool for selecting colour schemes for maps". In: *Cartographic Journal* 40.1 (2003), pp. 27–37. ISSN: 00087041. DOI: 10.1179/000870403235002042.

[HDK08] Björn Hartmann, Scott Doorley, and Scott R. Klemmer. "Hacking, mashing, gluing: Understanding opportunistic design". In: *Pervasive Computing* 7.3 (2008). DOI: 10.1109/MPRV.2008.54.

[Hee+08] Jeffrey Heer, Jock Mackinlay, Chris Stolte, and Maneesh Agrawala. "Graphical histories for visualization: supporting analysis, communication, and evaluation". In: *Transactions on Visualization and Computer Graphics* 14.6 (2008), pp. 1189–1196. DOI: 10.1109/tvcg.2008.137.

[HJ11] Charles D. Hansen and Chris R. Johnson, eds. *Visualization Handbook*. Elsevier Science, 2011. ISBN: 978-0123875822.

[HK12] Jiawei Han and Micheline Kamber. *Data mining: Concepts and techniques*. 3rd ed. Elsevier, 2012. ISBN: 9780123814791.

[HM90] Robert B Haber and David A McNabb. "Visualization idioms: A conceptual model for scientific visualization systems". In: *Visualization in scientific computing* 74 (1990), p. 93.

[HMS01] D. J. Hand, Heikki Mannila, and Padhraic Smyth. *Principles of data mining*. A Bradford book. The MIT Press, 2001. ISBN: 978-1-84628-765-7. DOI: 10.1007/978-1-84628-766-4.

[HMS03] Mark Hansen, Stuart Madnick, and Michael Siegel. "Data Integration Using Web Services". In: *Efficiency and effectiveness of xml tools and techniques and data integration over the web*. Ed. by Stéphane Bressan. Vol. 2590. Lecture Notes in Computer Science. Springer, 2003, pp. 165–182. ISBN: 978-3-540-00736-4. DOI: 10.1007/3-540-36556-7_15.

[Hor+18] Tom Horak, Sriram Karthik Badam, Niklas Elmqvist, and Raimund Dachselt. "When David Meets Goliath". In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. Ed. by Regan Mandryk. ACM Conferences. ACM, 2018, pp. 1–13. ISBN: 978-1-4503-5620-6. DOI: 10.1145/3173574.3173593.

[HR11] Frank Hardisty and Anthony C. Robinson. "The geoviz toolkit: using component-oriented coordination methods for geographic visualization and analysis". In: *International Journal of Geographical Information Science* 25.2 (2011), pp. 191–210. ISSN: 1365-8816. DOI: 10.1080/13658810903214203.

[HS] Marius Hogräfer and Hans-Jörg Schulz. *ReVize*. https://github.com/vis-au/revize. Last accessed 06-11-2024.

[HS19] Marius Hogräfer and Hans-Jörg Schulz. "ReVize: A Library for Visualization Toolchaining with Vega-Lite". In: *Proceedings of Smart Tools and Apps for Graphics (STAG)*. Eurographics Association, 2019, pp. 129–139. DOI: 10.2312/stag.20191375.

[Huy21] D. Huynh. *OpenRefine*. https://openrefine.org/. Last accessed 06-11-2024. 2021.

[HW10] Jeff Huang and Ryen W. White. "Parallel browsing behavior on the web". In: *Proceedings of the 21st conference on Hypertext and hypermedia*. Ed. by Mark Chignell. ACM, 2010, p. 13. ISBN: 978-1-4503-0041-4. DOI: 10.1145/1810617.1810622.

[ID91] Alfred Inselberg and Bernard Dimsdale. "Parallel Coordinates". In: *Human-Machine Interactive Systems*. Ed. by Allen Klinger. Languages and Information Systems. Springer, 1991, pp. 199–233. ISBN: 978-1-4684-5885-5. DOI: 10.1007/978-1-4684-5883-1{_}9.

[Inm05] William H. Inmon. *Building the data warehouse*. Fourth edition. Wiley technology publishing Timely, practical, reliable. Wiley, 2005. ISBN: 978-0-471-56960-2.

[Joh+16] Alistair E.W. Johnson, Tom J. Pollard, Lu Shen, Li-wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. "MIMIC-III, a freely accessible critical care database". In: *Scientific Data* 3.1 (2016), pp. 1–9. DOI: 10.1038/sdata.2016.35.

[Jud+04] Douglass Russell Judd, Bruce Karsh, Ram Subbaroyan, Troy Toman, Rahul Lahiri, and Patrick Lok. *Apparatus and method for searching and retrieving structured, semi-structured and unstructured content*. 2004.

[KC16] Sören Knudsen and Sheelagh Carpendale. "View Relations: An Exploratory Study on Between-View Meta-Visualizations". In: *Proceedings of the 9th Nordic Conference on Human-Computer Interaction*. Ed. by Staffan Björk and Eva Eriksson. ACM, 2016, pp. 1–10. ISBN: 978-1-4503-4763-1. DOI: 10.1145/2971485.2971566.

[Kei+06] Daniel A. Keim, Florian Mansmann, Jörn Schneidewind, and Hartmut Ziegler. "Challenges in Visual Data Analysis". In: *10th International Conference on Information Visualisation*. IEEE, 2006, pp. 9–16. DOI: 10.1109/iv.2006.31.

[Kei+08] Daniel Keim, Gennady Andrienko, Jean-Daniel Fekete, Carsten Görg, Jörn Kohlhammer, and Guy Melançon. "Visual Analytics: Definition, Process, and Challenges". In: *Information Visualization*. Ed. by David Hutchison, Jean-Daniel Fekete, Takeo Kanade, Andreas Kerren, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, and Chris North. Lecture Notes in Computer Science. Springer, 2008, pp. 154–175. ISBN: 978-3-540-70956-5. DOI: 10.1007/978-3-540-70956-5_7.

[Kei+10] Daniel Keim, Jörn Kohlhammer, Geoffrey Ellis, and Florian Mansmann, eds. *Mastering the information age: Solving problems with visual analytics*. Eurographics Association, 2010. ISBN: 9783905673777. DOI: 10.2312/14803.

[Kei01] Daniel A. Keim. "Visual exploration of large data sets". In: *Communications of the ACM* 44.8 (2001), pp. 38–44. ISSN: 0001-0782. DOI: 10.1145/381641.381656.

[Key+12] Alicia Key, Bill Howe, Daniel Perry, and Cecilia Aragon. "VizDeck". In: *Proceedings of the SIGMOD International Conference on Management of Data*. Ed. by K. Selçuk Candan. ACM, 2012, p. 681. ISBN: 978-1-4503-1247-9. DOI: 10.1145/2213836.2213931.

[KK17] N. Kerracher and J. Kennedy. "Constructing and Evaluating Visualisation Task Classifications: Process and Considerations". In: *Computer Graphics Forum* 36.3 (2017), pp. 47–59. ISSN: 01677055. DOI: 10.1111/cgf.13167.

[KKA95] D. A. Keim, H.-P. Kriegel, and M. Ankerst. "Recursive pattern: a technique for visualizing very large amounts of data". In: *Visualization '95*. Ed. by Gregory M. Nielson and Gregory M. Nielsen. IEEE, 1995, pp. 279–286. ISBN: 0-8186-7187-4. DOI: 10.1109/visual.1995.485140.

[KMT10] Daniel A. Keim, Florian Mansmann, and Jim Thomas. "Visual analytics: how much visualization and how much analytics?" In: *SIGKDD Explorations Newsletter* 11.2 (2010), pp. 5–8. ISSN: 1931-0145. DOI: 10.1145/1809400.1809403.

[KRL97] J. Kolojejchick, S. F. Roth, and P. Lucas. "Information appliances and tools in Visage". In: *Computer Graphics and Applications* 17.4 (1997), pp. 32–41. DOI: 10.1109/38.595266.

[KRM18] Guido Kraemer, Markus Reichstein, and Miguel D. Mahecha. "dimRed and coRanking—unifying dimensionality reduction in R". In: *R Journal* 10.1 (2018), pp. 342–358. ISSN: 2073-4859. DOI: 10.32614/rj-2018-039.

[Kus00] Nicholas Kushmerick. "Wrapper induction: Efficiency and expressiveness". In: *Artificial Intelligence* 118.1 (2000), pp. 15–68. ISSN: 0004-3702. DOI: 10.1016/s0004-3702(99)00100-9.

[KW02] Daniel A. Keim and Matthew O. Ward. "Visual Data Mining Techniques". In: *Intelligent Data Analysis: An Introduction*. Ed. by Michael Berthold. Springer, 2002, pp. 2–27.

[LC10] Liang-Yi Li and Gwo-Dong Chen. "A web browser interface to manage the searching and organizing of information on the web by learners". In: *Journal of Educational Technology & Society* 13.4 (2010). Last accessed 06-11-2024, pp. 86–97. ISSN: 1436-4522.

[Len11] Maurizio Lenzerini. "Ontology-based data management". In: *Proceedings of the 20th International Conference on Information and Knowledge Management*. Cikm '11. ACM, 2011, pp. 5–6. ISBN: 9781450307178. DOI: 10.1145/2063576.2063582.

[Lex+10] Alexander Lex, Mark Streit, Ernst Kruijff, and Dieter Schmalstieg. "Caleydo: Design and evaluation of a visual analysis framework for gene expression data in its biological context". In: *Proceedings of the Pacific Visualization Symposium (PacificVis)*. IEEE, 2010, pp. 57–64. DOI: 10.1109/pacificvis.2010.5429609.

[LH14] Zhicheng Liu and Jeffrey Heer. "The effects of interactive latency on exploratory visual analysis". In: *IEEE transactions on visualization and computer graphics* 20.12 (2014), pp. 2122–2131. DOI: 10.1109/tvcg.2014.2346452.

[Lie+11] Michael D. Lieberman, Sima Taheri, Huimin Guo, Fatemeh Mirrashed, Inbal Yahav, Aleks Aris, and Ben Shneiderman. "Visual exploration across biomedical databases". In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 8.2 (2011), pp. 536–550. DOI: 10.1109/tcbb.2010.1.

[LMM06] Wei Liu, Xiaofeng Meng, and Weiyi Meng. "Vision-based web data records extraction". In: *Proc. 9th international workshop on the web and databases*. Last accessed 06-11-2024. 2006, pp. 20–25.

[Lo +19] Paolo Lo Giudice, Lorenzo Musarella, Giuseppe Sofo, and Domenico Ursino. "An approach to extracting complex knowledge patterns among concepts belonging to structured, semi-structured and unstructured sources in a data lake". In: *Information Sciences* 478 (2019), pp. 606–626. ISSN: 00200255. DOI: 10.1016/j.ins.2018.11.052.

[Lop+10] Christian T. Lopes, Max Franz, Farzana Kazi, Sylva L. Donaldson, Quaid Morris, and Gary D. Bader. "Cytoscape Web: an interactive web-based network browser". In: *Bioinformatics* 26.18 (2010), pp. 2347–2348. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btq430.

[Lot19] Alexander Loth. *Visual analytics with Tableau*. John Wiley & Sons, 2019. ISBN: 1119560225. DOI: 10.1002/9781119561996.

[LP95] Edward A. Lee and T. M. Parks. "Dataflow process networks". In: *Proceedings of the IEEE* 83.5 (1995), pp. 773–801. ISSN: 00189219. DOI: 10.1109/5.381846.

[LSW98] Ora Lassila, Ralph R. Swick, and World Wide Web Consortium. *Resource Description Framework (RDF) Model and Syntax Specification*. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.44.6030. 1998.

[Lud+06] Bertram Ludäscher, Ilkay Altintas, Chad Berkley, Dan Higgins, Efrat Jaeger, Matthew Jones, Edward a. Lee, Jing Tao, and Yang Zhao. "Scientific workflow management and the Kepler system". In: *Concurrency and Computation: Practice and Experience* 18.10 (2006), pp. 1039–1065. DOI: 10.1002/cpe.994.

[LWH14] Chen Liu, Jianwu Wang, and Yanbo Han. "Mashroom+: An Interactive Data Mashup Approach with Uncertainty Handling". In: *Journal of Grid Computing* 12.2 (2014), pp. 221–244. ISSN: 1570-7873. DOI: 10.1007/s10723-013-9280-5.

[Mai06] Oded Z. Maimon, ed. *Data mining and knowledge discovery handbook*. 2. ed. Springer, 2006. ISBN: 978-0-387-25465-4.

[Maj+04] Shalil Majithia, Matthew Shields, Ian Taylor, and Ian Wang. "Triana: A graphical web service composition and execution toolkit". In: *Web Services, 2004. Proceedings. IEEE International Conference on*. IEEE, 2004, pp. 514–521. ISBN: 0-7695-2167-3. DOI: 10.1109/ICWS.2004.1314777.

[Mar+14] Marc Streit, Alexander Lex, Samuel Gratzl, Christian Partl, Dieter Schmalstieg, Hanspeter Pfister, Peter J Park, and Nils Gehlenborg. "Guided visual exploration of genomic stratifications in cancer". In: *Nature Methods* 11.9 (2014), pp. 884–885. ISSN: 1548-7105. DOI: 10.1038/nmeth.3088.

[Max+07] E. Michael Maximilien, Hernan Wilkinson, Nirmit Desai, and Stefan Tai. "A Domain-Specific Language for Web APIs and Services Mashups". In: *Service-Oriented Computing – ICSOC 2007*. Vol. 4749. Lecture Notes in Computer Science. Springer, 2007, pp. 13–26. ISBN: 978-3-540-74973-8. DOI: 10.1007/978-3-540-74974-5_2.

[MBE03] Brahim Medjahed, Athman Bouguettaya, and Ahmed K. Elmagarmid. "Composing Web services on the Semantic Web". In: *The VLDB Journal The International Journal on Very Large Data Bases* 12.4 (2003), pp. 333–351. ISSN: 1066-8888. DOI: 10.1007/s00778-003-0101-5.

[Mel+06] Alexandra Meliou, David Chu, Joseph Hellerstein, Carlos Guestrin, and Wei Hong. "Data gathering tours in sensor networks". In: *Ispn 2006*. Ed. by John Stankovic, Phillip Gibbons, Stephen Wicker, and Joe Paradiso. ACM, 2006, p. 43. ISBN: 1595933344. DOI: 10.1145/1127777.1127788.

[MHS07] Jock Mackinlay, Pat Hanrahan, and Chris Stolte. "Show me: automatic presentation for visual analysis". In: *Transactions on Visualization and Computer Graphics* 13.6 (2007), pp. 1137–1144. DOI: 10.1109/tvcg.2007.70594.

[MM04a] Matteo Magnani and Danilo Montesi. "A unified approach to structured, semistructured and unstructured data". In: *University of Bologna, Department of Computer Science, Tech. Rep* 9 (2004), p. 2004.

[MM04b] N. Milanovic and M. Malek. "Current solutions for Web service composition". In: *Internet Computing* 8.6 (2004), pp. 51–59. ISSN: 1089-7801. DOI: 10.1109/mic.2004.58.

[MMM+04] Frank Manola, Eric Miller, Brian McBride, et al. "RDF primer". In: *W3C recommendation* 10.1-107 (2004), p. 6.

[Mor+13] Jeffrey T. Morisette, Catherine S. Jarnevich, Tracy R. Holcombe, Colin B. Talbert, Drew Ignizio, Marian K. Talbert, Claudio Silva, David Koop, Alan Swanson, and Nicholas E. Young. "VisTrails SAHM: visualization and workflow management for species habitat modeling". In: *Ecography* 36.2 (2013), pp. 129–135. ISSN: 09067590. DOI: 10.1111/j.1600-0587.2012.07815.x.

[Mor13] Kenneth Moreland. "A survey of visualization pipelines". In: *Transactions on Visualization and Computer Graphics* 19.3 (2013), pp. 367–378. DOI: 10.1109/tvcg.2012.133.

[Mou+12] Anastasia Moumtzidou, Jaakko Kukkonen, Victor Epitropou, Stefanos Vrochidis, Sascha Voth, Anastasios Bassoukos, Kostas Karatzas, Jürgen Moßgraber, Ioannis Kompatsiaris, and Ari Karppinen. "Environmental data extraction from multimedia resources". In: *Proceedings of the 1st international workshop on Multimedia analysis for ecological data*. Ed. by Concetto Spampinato. ACM, 2012, p. 13. ISBN: 9781450315883. DOI: 10.1145/2390832.2390836.

[MR06] Zdravko Markov and Ingrid Russell. "An introduction to the WEKA data mining system". In: *SIGCSE Bulletin* 38.3 (2006), pp. 367–368. ISSN: 0097-8418. DOI: 10.1145/1140123.1140127.

[Mül18] Felix Müller. *Data extraction engine for structured, semi-structured and unstructured data with automated labeling and classification of data patterns or data elements therein, and corresponding method thereof*. US Patent App. 15/387,070. 2018.

[Mun09] Tamara Munzner. "A nested model for visualization design and validation". In: *Transactions on Visualization and Computer Graphics* 15.6 (2009), pp. 921–928. DOI: 10.1109/tvcg.2009.111.

[Mun15] Tamara Munzner. *Visualization analysis & design*. AK Peters Visualization series. CRC Press, 2015. ISBN: 9781466508934.

[Mvv99] Jurriaan D. Mulder, Jarke J. van Wijk, and Robert van Liere. "A survey of computational steering environments". In: *Future Generation Computer Systems* 15.1 (1999), pp. 119–129. ISSN: 0167-739x. DOI: 10.1016/s0167-739x(98)00047-8.

[Nat98] David A. Nation. "WebTOC". In: *CHI Conference Summary on Human Factors in Computing Systems*. Ed. by Clare-Marie Karat. ACM, 1998, pp. 185–186. ISBN: 978-1-58113-028-7. DOI: 10.1145/286498.286664.

[Nona] Lars Nonnemann. *AnyProc*. https://github.com/nonnemann/AnyProc. Last accessed 06-11-2024.

[Nonb] Lars Nonnemann. *UniVA*. https://github.com/nonnemann/UnIVA. Last accessed 06-11-2024.

[Non+19] Lars Nonnemann, Marian Haescher, Mario Aehnelt, Gerald Bieber, Holger Diener, and Bodo Urban. "HealthHand A Visual Interface for eHealth Monitoring". In: *Symposium on Computers and Communications (ISCC)*. IEEE, 2019, pp. 1093–1096. ISBN: 978-1-7281-2999-0. DOI: 10.1109/iscc47284.2019.8969647.

[Non+20] Lars Nonnemann, Heidrun Schumann, Bodo Urban, Mario Aehnelt, and Hans-Jörg Schulz. "A Characterization of Data Exchange between Visual Analytics Tools". In: *24th International Conference Information Visualisation (IV)*. IEEE, 2020, pp. 368–377. ISBN: 978-1-7281-9134-8. DOI: 10.1109/iv51561.2020.00066.

[Non+21] Lars Nonnemann, Marius Hogräfer, Heidrun Schumann, Bodo Urban, and Hans-Jörg Schulz. "Customizable Coordination of Independent Visual Analytics Tools". In: *International Workshop on Visual Analytics (EuroVA) at EuroVis*. The Eurographics Association, 2021, pp. 25–29. ISBN: 978-3-03868-150-2. DOI: 10.2312/eurova.20211094.

[Non+22] Lars Nonnemann, Marius Hogräfer, Martin Röhlig, Heidrun Schumann, Bodo Urban, and Hans-Jörg Schulz. "A Data-Driven Platform for the Coordination of Independent Visual Analytics Tools". In: *Computers & Graphics* 106 (2022), pp. 152–160. ISSN: 00978493. DOI: 10.1016/j.cag.2022.05.023.

[Nor+02] Chris North, Nathan Conklin, Kiran Indukuri, and Varun Saini. "Visualization Schemas and a Web-Based Architecture for Custom Multiple-View Visualization of Multiple-Table Databases". In: *Information Visualization* 1.3-4 (2002), pp. 211–228. ISSN: 1473-8716. DOI: 10.1057/palgrave.ivs.9500020.

[Nor+03] Chris North, Nathan Conklin, Kiran Indukuri, Varun Saini, and Qiang Yu. "Fusion: Interactive coordination of diverse data, visualizations, and mining algorithms". In: *CHI Extended Abstracts on Human Factors in Computing Systems*. Ed. by Gilbert Cockton. ACM, 2003, p. 626. ISBN: 978-1-58113-637-1. DOI: 10.1145/765891.765897.

[Nor+11] Chris North, Remco Chang, Alex Endert, Wenwen Dou, Richard May, Bill Pike, and Glenn Fink. "Analytic provenance". In: *CHI '11 Extended Abstracts on Human Factors in Computing Systems*. Ed. by Desney Tan. ACM Digital Library. ACM, 2011, p. 33. ISBN: 978-1-4503-0268-5. DOI: 10.1145/1979742.1979570.

[NS00] Chris North and Ben Shneiderman. "Snap-together visualization". In: *Proceedings of the working conference on Advanced visual interfaces*. ACM, 2000, pp. 128–135. ISBN: 978-1-58113-252-6. DOI: 10.1145/345513.345282.

[Oin+04] Tom Oinn, Matthew Addis, Justin Ferris, Darren Marvin, Martin Senger, Mark Greenwood, Tim Carver, Kevin Glover, Matthew R. Pocock, Anil Wipat, et al. "Taverna: a tool for the composition and enactment of bioinformatics workflows". In: *Bioinformatics (Oxford, England)* 20.17 (2004), pp. 3045–3054. DOI: 10.1093/bioinformatics/bth361.

[Par+15] Mario Andrés Paredes-Valverde, Giner Alor-Hernández, Alejandro Rodríguez-González, Rafael Valencia-García, and Enrique Jiménez-Domingo. "A systematic review of tools, languages, and methodologies for mashup development". In: *Software: Practice and Experience* 45.3 (2015), pp. 365–397. ISSN: 00380644. DOI: 10.1002/spe.2233.

[Pel03] C. Peltz. "Web services orchestration and choreography". In: *Computer* 36.10 (2003), pp. 46–52. ISSN: 0018-9162. DOI: 10.1109/mc.2003.1236471.

[PJ95] Steven G. Parker and Christopher R. Johnson. "SCIRun". In: *1995 International Conference on Supercomputing (SC95)*. Ed. by Sid Karin. ACM, 1995, 52–es. ISBN: 0897918169. DOI: 10.1145/224170.224354.

[PMH18] Jorge Poco, Angela Mayhua, and Jeffrey Heer. "Extracting and Retargeting Color Mappings from Bitmap Images of Visualizations". In: *Transactions on Visualization and Computer Graphics* 24.1 (2018), pp. 637–646. ISSN: 1941-0506. DOI: 10.1109/tvcg.2017.2744320.

[PND10] Stefan Pietschmann, Tobias Nestler, and Florian Daniel. "Application composition at the presentation layer". In: *Proceedings of the 12th International Conference on Information Integration and Web-based Applications & Services*. Ed. by Eric Pardede. ACM, 2010, p. 461. ISBN: 978-1-4503-0421-4. DOI: 10.1145/1967486.1967558.

[Pra+18] Ruby Kala Prakasam, Aline Götze, Sophie von Keyserlingk, Anselm Jünemann, Martin Röhlig, Oliver Stachs, and Dagmar-Christiane Fischer. "Spectral-Domain Optical Coherence Tomography for Determination of Retinal Thickness in Pediatric Patients with Mild-To-Moderate Chronic Kidney Disease: A Cross-Sectional Study". In: *Current Eye Research* 44.2 (2018), pp. 206–211. DOI: 10.1080/02713683.2018.1522649.

[Pra+19] Ruby Kala Prakasam, Martin Röhlig, Dagmar-Christiane Fischer, Aline Götze, Anselm Jünemann, Heidrun Schumann, and Oliver Stachs. "Deviation maps for understanding thickness changes of inner retinal layers in children with type 1 diabetes mellitus". In: *Curr Eye Res* 44.7 (2019), pp. 746–752. DOI: 10.1080/02713683.2019.1591463.

[Pra+20] Ruby Kala Prakasam, Aleksandra Matuszewska-Iwanicka, Dagmar-Christiane Fischer, Heidrun Schumann, Diethelm Tschöpe, Bernd Stratmann, Hans-Joachim Hettlich, Rudolf F. Guthoff, Oliver Stachs, and Martin Röhlig. "Thickness of intraretinal layers in patients with type 2 diabetes mellitus depending on a concomitant diabetic neuropathy: results of a cross-sectional study using deviation maps for OCT data analysis". In: *Biomedicines* 8.7 (2020). DOI: 10.3390/biomedicines8070190.

[Pru07] Mark Pruett. *Yahoo! pipes*. O'Reilly, 2007. ISBN: 9780596514532.

[PS+06] Eric Prud, Andy Seaborne, et al. *SPARQL query language for RDF*. Tech. rep. Last Accessed: 06-11-2024. Technical report, W3C, 2006.

[Pv07] Mike P. Papazoglou and Willem-Jan van den Heuvel. "Service oriented architectures: approaches, technologies and research issues". In: *The VLDB Journal* 16.3 (2007), pp. 389–415. ISSN: 10668888. DOI: 10.1007/s00778-007-0044-3.

[PWJ97] Steven G. Parker, David M. Weinstein, and Christopher R. Johnson. "The SCIRun Computational Steering Software System". In: *Modern software tools for scientific computing*. Ed. by Erlend Arge, A. M. Bruaset, and Hans Petter Langtangen. Springer Science+Business Media, 1997, pp. 5–44. ISBN: 978-1-4612-7368-4. DOI: 10.1007/978-1-4612-1986-6_1.

[PZL08] Cesare Pautasso, Olaf Zimmermann, and Frank Leymann. "Restful web services vs. big web services". In: *Proceedings of the 17th international conference on World Wide Web*. Ed. by Jinpeng Huai. ACM, 2008, p. 805. ISBN: 9781605580852. DOI: 10.1145/1367497.1367606.

[Rac12] Jeffrey S. Racine. "R-Studio: A platform-independent IDE for R and Sweave". In: *Journal of Applied Econometrics* 27.1 (2012), pp. 167–172. ISSN: 08837252.

[Rad+15] Axel Radloff, Christian Tominski, Thomas Nocke, and Heidrun Schumann. "Supporting presentation and discussion of visualization results in smart meeting rooms". In: *The Visual Computer* 31.9 (2015), pp. 1271–1286. ISSN: 0178-2789. DOI: 10.1007/s00371-014-1010-x.

[RBP15] Petar Ristoski, Christian Bizer, and Heiko Paulheim. "Mining the Web of Linked Data with RapidMiner". In: *Journal of Web Semantics* 35 (2015), pp. 142–151. ISSN: 1570-8268. DOI: 10.1016/j.websem.2015.06.004.

[RGF11] Michael Rooke, Tovi Grossman, and George Fitzmaurice. "AppMap: Exploring User Interface Visualizations". In: *Proceedings of Graphics Interface*. 2011. ISBN: 978-1-4503-0693-5. DOI: 10.5555/1992917.1992936.

[RLS11] Axel Radloff, Martin Luboschik, and Heidrun Schumann. "Smart Views in Smart Environments". In: *Smart graphics*. Ed. by Lutz Dickmann, Gerald Volkmann, Rainer Malaka, Susanne Boll, Antonio Krüger, and Patrick Olivier. Lecture Notes in Computer Science. Springer, 2011, pp. 1–12. ISBN: 978-3-642-22571-0.

[RM11] Bernice E. Rogowitz and Naim Matasci. "Metadata Mapper: a web service for mapping data between independent visual analysis components, guided by perceptual rules". In: *Human Vision and Electronic Imaging XVI*. SPIE Proceedings. SPIE, 2011, p. 78650i. DOI: 10.1117/12.881734.

[Rob07] Jonathan C. Roberts. "State of the Art: Coordinated & Multiple Views in Exploratory Visualization". In: *Proceedings / Fifth International Conference on Coordinated & Multiple Views in Exploratory Visualization, CMV 2007*. Ed. by Gennady Andrienko, Jonathan C. Roberts, and Chris Weaver. IEEE, 2007, pp. 61–71. ISBN: 0-7695-2903-8. DOI: 10.1109/cmv.2007.20.

[Rod23] Robert N. Rodriguez. "Building regression models with SAS". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 3.1 (2023), pp. 1–11.

[Röh+18] Martin Röhlig, Christoph Schmidt, Ruby Kala Prakasam, Heidrun Schumann, and Oliver Stachs. "Visual Analysis of Retinal Changes with Optical Coherence Tomography". In: *Vis Comput* 34.9 (2018), pp. 1209–1224. DOI: 10.1007/s00371-018-1486-x.

[Röh+19] Martin Röhlig, Ruby Kala Prakasam, Jörg Stüwe, Christoph Schmidt, Oliver Stachs, and Heidrun Schumann. "Enhanced Grid-Based Visual Analysis of Retinal Layer Thickness with Optical Coherence Tomography". In: *Information* 10.9 (2019), p. 266. DOI: 10.3390/info10090266.

[Röh+23] Martin Röhlig, Lars Nonnemann, Hans-Jörg Schulz, Oliver Stachs, and Heidrun Schumann. "Towards a Unified User Interface for Visual Analysis of Retinal Data in Ophthalmology". In: *CoRR* abs/1912.08558 (2023). In preparation as journal article for Computer Graphics Forum. DOI: 10.48550/arXiv.2301.01840.

[RS09] René Rosenbaum and Heidrun Schumann. "Progressive refinement: more than a means to overcome limited bandwidth". In: *Visualization and Data Analysis*. Ed. by Katy Börner and Jinah Park. SPIE Proceedings. SPIE, 2009. DOI: 10.1117/12.810501.

[Run+13] Atul Rungta, Brian Summa, Dogan Demir, Peer-Timo Bremer, and Valerio Pascucci. "ManyVis: Multiple applications in an integrated visualization environment". In: *Transactions on Visualization and Computer Graphics* 19.12 (2013), pp. 2878–2885. DOI: 10.1109/tvcg.2013.174.

[Sac+14] Dominik Sacha, Andreas Stoffel, Florian Stoffel, Bum Chul Kwon, Geoffrey Ellis, and Daniel A. Keim. "Knowledge Generation Model for Visual Analytics". In: *Transactions on Visualization and Computer Graphics* 20.12 (2014), pp. 1604–1613. DOI: 10.1109/tvcg.2014.2346481.

[Sac+17] Dominik Sacha, Leishi Zhang, Michael Sedlmair, John A. Lee, Jaakko Peltonen, Daniel Weiskopf, Stephen C. North, and Daniel A. Keim. "Visual Interaction with Dimensionality Reduction: A Structured Literature Analysis". In: *Transactions on Visualization and Computer Graphics* 23.1 (2017), pp. 241–250. ISSN: 1941-0506. DOI: 10.1109/tvcg.2016.2598495.

[San+09] Emanuele Santos, Lauro Lins, James Ahrens, Juliana Freire, and Cláudio Silva. "VisMashup: Streamlining the creation of custom visualization applications". In: *Transactions on Visualization and Computer Graphics* 15.6 (2009), pp. 1539–1546. DOI: 10.1109/tvcg.2009.195.

[Sat+17] Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. "Vega-Lite: A Grammar of Interactive Graphics". In: *Transactions on Visualization and Computer Graphics* 23.1 (2017), pp. 341–350. DOI: 10.1109/tvcg.2016.2599030.

[Sch+13] Hans-Jörg Schulz, Thomas Nocke, Magnus Heitzler, and Heidrun Schumann. "A Design Space of Visualization Tasks". In: *Transactions on Visualization and Computer Graphics* 19.12 (2013), pp. 2366–2375. ISSN: 1941-0506. DOI: 10.1109/tvcg.2013.120.

[Sch+16] Hans-Jörg Schulz, Marco Angelini, Giuseppe Santucci, and Heidrun Schumann. "An Enhanced Visualization Process Model for Incremental Visualization". In: *Transactions on Visualization and Computer Graphics* 22.7 (2016), pp. 1830–1842. DOI: 10.1109/tvcg.2015.2462356.

[Sch+17] Hans-Jörg Schulz, Thomas Nocke, Magnus Heitzler, and Heidrun Schumann. "A systematic view on data descriptors for the visual analysis of tabular data". In: *Information Visualization* 16.3 (2017), pp. 232–256. ISSN: 1473-8716. DOI: 10.1177/1473871616667767.

[Sch+19a] Christoph Schmidt, Martin Röhlig, Bastian Grundel, Philipp Daumke, Marc Ritter, Andreas Stahl, Paul Rosenthal, and Heidrun Schumann. "Combining Visual Cleansing and Exploration for Clinical Data". In: *Proceedings of the Workshop on Visual Analytics in Healthcare (VAHC)*. IEEE, 2019, pp. 25–32. DOI: 10.1109/vahc47919.2019.8945034.

[Sch+19b] Hans-Jörg Schulz, Martin Röhlig, Lars Nonnemann, Mario Aehnelt, Holger Diener, Bodo Urban, and Heidrun Schumann. "Lightweight Coordination of Multiple Independent Visual Analytics Tools". In: *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science and Technology Publications, 2019, pp. 106–117. ISBN: 978-989-758-354-4. DOI: 10.5220/0007571101060117.

[Sch+20] Hans-Jörg Schulz, Martin Röhlig, Lars Nonnemann, Marius Hogräfer, Mario Aehnelt, Bodo Urban, and Heidrun Schumann. "A Layered Approach to Lightweight Toolchaining in Visual Analytics". In: *Computer Vision, Imaging and Computer Graphics Theory and Applications*. Ed. by Ana Paula Cláudio. Vol. 1182. Communications in Computer and Information Science. Springer International Publishing, 2020, pp. 313–337. ISBN: 978-3-030-41589-1. DOI: 10.1007/978-3-030-41590-7_13.

[Sch+21] Christoph Schmidt, Bastian Grundel, Heidrun Schumann, and Paul Rosenthal. "Annotations in Different Steps of Visual Analytics". In: *Proceedings of the International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (IVAPP)*. Insticc. SciTePress, 2021, pp. 155–163. DOI: 10.5220/0010198001550163.

[Sch67] Richard P. Schoemaker. "Visual Path Analysis". In: *Journal of Petroleum Technology* 19.02 (1967), pp. 177–185. ISSN: 0149-2136. DOI: 10.2118/1596-pa.

[SGS04] D. Skogan, R. Gronmo, and I. Solheim. "Web service composition in UML". In: *Proceedings / Eighth International Enterprise Distributed Object Computing Conference, 2004, EDOC 2004*. IEEE, 2004, pp. 47–57. ISBN: 0-7695-2214-9. DOI: 10.1109/edoc.2004.1342504.

[Sha+03] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S. Baliga, Jonathan T. Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. "Cytoscape: a software environment for integrated models of biomolecular interaction networks". In: *Genome research* 13.11 (2003). ISSN: 1088-9051. DOI: 10.1101/gr.1239303.

[She+02] Quan Z. Sheng, Boualem Benatallah, Marlon Dumas, and Eileen Oi-Yan Mak. "Self-serv: A Platform for Rapid Composition of Web Services in a Peer-to-Peer Environment". In: *Proceedings 2002 VLDB Conference*. Elsevier textbooks, 2002, pp. 1051–1054. ISBN: 9781558608696. DOI: 10.1016/b978-155860869-6/50106-2.

[She+14] Quan Z. Sheng, Xiaoqiang Qiao, Athanasios V. Vasilakos, Claudia Szabo, Scott Bourne, and Xiaofei Xu. "Web services composition: A decade's overview". In: *Information Sciences* 280 (2014), pp. 218–238. ISSN: 00200255. DOI: 10.1016/j.ins.2014.04.054.

[She99] Amit P. Sheth. "Changing Focus on Interoperability in Information Systems:From System, Syntax, Structure to Semantics". In: *Interoperating geographic information systems*. Ed. by Michael F. Goodchild. Springer Science+Media Business, 1999, pp. 5–29. ISBN: 978-1-4613-7363-6. DOI: 10.1007/978-1-4615-5189-8{_}2.

[Shn03] Ben Shneiderman. "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations". In: *The Craft of Information Visualization*. Ed. by Benjamin Bederson. Morgan Kaufmann and Safari, 2003, pp. 364–371. ISBN: 9781558609150. DOI: 10.1016/b978-155860915-0/50046-9.

[Shn92] Ben Shneiderman. "Tree visualization with tree-maps". In: *Transactions on Graphics* 11.1 (1992), pp. 92–99. ISSN: 07300301. DOI: 10.1145/102377.115768.

[Shn94] Ben Shneiderman. "Dynamic queries for visual information seeking". In: *IEEE software* 11.6 (1994), pp. 70–77. DOI: doi:10.1109/52.329404.

[SL90] Amit P. Sheth and James A. Larson. "Federated database systems for managing distributed, heterogeneous, and autonomous databases". In: *Computing Surveys (CSUR)* 22.3 (1990), pp. 183–236.

[SLM04] Will J. Schroeder, Bill Lorensen, and Ken Martin. *The visualization toolkit: an object-oriented approach to 3D graphics*. Kitware, 2004. ISBN: 1930934122.

[SM00] Heidrun Schumann and Wolfgang Müller. *Visualisierung: Grundlagen und allgemeine Methoden*. Springer, 2000. ISBN: 9783642571930. DOI: 10.1007/978-3-642-57193-0.

[SM09] Marc H. Scholl and Svetlana Mansmann. "Visual On-Line Analytical Processing (OLAP)". In: *Encyclopedia of Database Systems*. Springer US, 2009, pp. 3388–3395. ISBN: 978-0-387-39940-9. DOI: 10.1007/978-0-387-39940-9_447.

[Spe14] Robert Spence. *Information Visualization: An Introduction*. 3rd ed. 2014. Springer eBook Collection Computer Science. Springer, 2014. ISBN: 9783319073415. DOI: 10.1007/978-3-319-07341-5.

[SPG08] Yogesh L. Simmhan, Beth Plale, and Dennis Gannon. "Karma2". In: *International Journal of Web Services Research* 5.2 (2008), pp. 1–22. ISSN: 1545-7362. DOI: 10.4018/jwsr.2008040101.

[SPG14] Charles D. Stolper, Adam Perer, and David Gotz. "Progressive Visual Analytics: User-Driven Visual Exploration of In-Progress Analytics". In: *Transactions on Visualization and Computer Graphics* 20.12 (2014), pp. 1653–1662. DOI: 10.1109/tvcg.2014.2346574.

[SPM04] J. A. Sanchez, C. Proal, and F. Maldonao-Naude. "Supporting structured, semi-structured and unstructured data in digital libraries". In: *Proceedings of the Fifth Mexican International Conference in Computer Science*. Ed. by Ricardo Baeza-Yates. IEEE, 2004, pp. 368–375. ISBN: 0-7695-2160-6. DOI: 10.1109/enc.2004.1342629.

[Spo07] Anselm Spoerri. "Visual Mashup of Text and Media Search Results". In: *11th International Conference Information Visualization*. Ed. by Ebad Banissi. IEEE Computer Society, 2007, pp. 216–221. ISBN: 0-7695-2900-3. DOI: 10.1109/iv.2007.125.

[SRJ18] Kuldeep Sambrekar, Vijay. S. Rajpurohit, and Jui Joshi. "A Proposed Technique for Conversion of Unstructured Agro-Data to Semi-Structured or Structured Data". In: *Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*. IEEE, 2018, pp. 1–5. ISBN: 978-1-5386-5257-2. DOI: 10.1109/iccubea.2018.8697432.

[SRS18] Christoph Schmidt, Paul Rosenthal, and Heidrun Schumann. "Annotations as a Support for Knowledge Generation - Supporting Visual Analytics in the Field of Ophthalmology". In: *Proceedings of the International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (IVAPP)*. Insticc. SciTePress, 2018, pp. 264–272. DOI: 10.5220/0006615902640272.

[SRS20] Christoph Schmidt, Paul Rosenthal, and Heidrun Schumann. "Varying Annotations in the Steps of the Visual Analysis". In: *CoRR* abs/2008.08806 (2020). DOI: 10.48550/arXiv.2008.08806.

[STH02] C. Stolte, D. Tang, and P. Hanrahan. "Polaris: a system for query, analysis, and visualization of multidimensional relational databases". In: *Transactions on Visualization and Computer Graphics* 8.1 (2002), pp. 52–65. ISSN: 1077-2626. DOI: 10.1109/2945.981851.

[Str+09] Marc Streit, Alexander Lex, Michael Kalkusch, Kurt Zatloukal, and Dieter Schmalstieg. "Caleydo: connecting pathways and gene expression". In: *Bioinformatics (Oxford, England)* 25.20 (2009), pp. 2760–2761. DOI: 10.1093/bioinformatics/btp432.

[Str+12] Marc Streit, Hans-Jörg Schulz, Alexander Lex, Dieter Schmalstieg, and Heidrun Schumann. "Model-driven design for the visual analysis of heterogeneous data". In: *Transactions on Visualization and Computer Graphics* 18.6 (2012), pp. 998–1010. DOI: 10.1109/tvcg.2011.108.

[Stu+06] Wolfgang Stuerzlinger, Olivier Chapuis, Dusty Phillips, and Nicolas Roussel. "User interface façades". In: *Proceedings of the 19th annual symposium on User interface software and technology*. Ed. by Pierre Wellner. ACM, 2006, p. 309. ISBN: 1595933131. DOI: 10.1145/1166253.1166301.

[Tan+20] Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, and Vipin Kumar. *Introduction to data mining*. Second edition, global edition. Pearson, 2020. ISBN: 978-0-273-77532-4.

[TBB10] Dan Tenenbaum, J. Christopher Bare, and Nitin S. Baliga. "GTC: A web server for integrating systems biology data with web tools and desktop applications". In: *Source code for biology and medicine* 5 (2010), p. 7. DOI: 10.1186/1751-0473-5-7.

[TBT15] M. B. Taylor, T. Boch, and J. Taylor. "SAMP, the Simple Application Messaging Protocol: Letting applications talk to each other". In: *Astronomy and Computing* 11 (2015), pp. 81–90. ISSN: 2213-1337. DOI: 10.1016/j.ascom.2014.12.007.

[Tea] R Core Team. *R: A Language and Environment for Statistical Computing*. https://www.r-project.org/. Last Accessed: 06-11-2024. R Foundation for Statistical Computing.

[TG02] Masahiro Takatsuka and Mark Gahegan. "GeoVISTA Studio: a codeless visual programming environment for geoscientific data analysis and visualization". In: *Computers & Geosciences* 28.10 (2002), pp. 1131–1144. ISSN: 0098-3004. DOI: 10.1016/s0098-3004(02)00031-6.

[Thu+09] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff, and Raghotham Murthy. "Hive: a warehousing solution over a map-reduce framework". In: *Proceedings of the VLDB Endowment* 2.2 (2009), pp. 1626–1629. ISSN: 21508097. DOI: 10.14778/1687553.1687609.

[TIC09] Matthew Tobiasz, Petra Isenberg, and Sheelagh Carpendale. "Lark: Coordinating co-located collaboration with information visualization". In: *Transactions on Visualization and Computer Graphics* 15.6 (2009), pp. 1065–1072. DOI: 10.1109/tvcg.2009.162.

[TMC04] Desney S. Tan, Brian Meyers, and Mary Czerwinski. "WinCuts: Manipulating arbitrary window regions for more effective use of screen space". In: *CHI Extended Abstracts on Human Factors in Computing Systems*. Ed. by Elizabeth Dykstra-Erickson. ACM, 2004, p. 1525. ISBN: 1581137036. DOI: 10.1145/985921.986106.

[Tom+06] Christian Tominski, James Abello, Frank van Ham, and Heidrun Schumann. "Fisheye Tree Views and Lenses for Graph Visualization". In: *Tenth International Conference on Information Visualization, 2006. IV 2006*. IEEE, 2006, pp. 17–24. ISBN: 0-7695-2602-0. DOI: 10.1109/iv.2006.54.

[Tom+21] Christian Tominski, Gennady L. Andrienko, Natalia V. Andrienko, Susanne Bleisch, Sara Irina Fabrikant, Eva Mayr, Silvia Miksch, Margit Pohl, and André Skupin. "Toward flexible visual analytics augmented through smooth display transitions". In: *Visual Informatics* 5.3 (2021), pp. 28–38. DOI: 10.1016/j.visinf.2021.06.004.

[TS20] Christian Tominski and Heidrun Schumann. *Interactive visual data analysis*. AK Peters Visualization series. CRC Press LLC, 2020. ISBN: 9781315152707. DOI: 10.1201/9781315152707.

[Tuk08] John W. Tukey. *Exploratory Data Analysis*. Springer New York, 2008, pp. 192–194. ISBN: 978-0-387-32833-1. DOI: 10.1007/978-0-387-32833-1_136.

[Ull97] Jeffrey D. Ullman. "Information integration using logical views". In: *Database theory - ICDT '97*. Ed. by Foto N. Afrati and Phokion Kolaitis. Vol. 1186. Lecture Notes in Computer Science. Springer, 1997, pp. 19–40. ISBN: 978-3-540-62222-2. DOI: 10.1007/3-540-62222-5{_}34.

[van+03] van Der Aalst, Wil MP, Arthur H. M. Ter Hofstede, Bartek Kiepuszewski, and Alistair P. Barros. "Workflow Patterns". In: *Distributed and Parallel Databases* 14.1 (2003), pp. 5–51. ISSN: 0926-8782.

[van05] J. J. van Wijk. "The Value of Visualization". In: *Visualization 2005*. Ed. by Cláudio T. Silva. IEEE, 2005, pp. 79–86. ISBN: 0-7803-9462-3. DOI: 10.1109/visual.2005.1532781.

[Vie+07] Fernanda B. Viegas, Martin Wattenberg, Frank van Ham, Jesse Kriss, and Matt McKeon. "ManyEyes: a site for visualization at internet scale". In: *Transactions on Visualization and Computer Graphics* 13.6 (2007), pp. 1121–1128. ISSN: 1077-2626. DOI: 10.1109/tvcg.2007.70577.

[Vo+11] Huy T. Vo, Joao L.D. Comba, Berk Geveci, and Claudio T. Silva. "Streaming-Enabled Parallel Data Flow Framework in the Visualization ToolKit". In: *Computing in Science & Engineering* 13.5 (2011), pp. 72–83. ISSN: 1521-9615. DOI: 10.1109/mcse.2011.88.

[VR17] Elio Ventocilla and Maria Riveiro. "Visual Analytics Solutions as 'off-the-Shelf' Libraries". In: *21st International Conference Information Visualisation (IV)*. IEEE, 2017, pp. 281–287. ISBN: 978-1-5386-0831-9. DOI: 10.1109/iV.2017.77.

[Wal+10] Manuela Waldner, Werner Puff, Alexander Lex, Marc Streit, and Dieter Schmalstieg. "Visual links across applications". In: *Proceedings of Graphics Interface 2010*. 2010, pp. 129–136.

[War12] Wendy A. Warr. "Scientific workflow systems: Pipeline Pilot and KNIME". In: *Journal of computer-aided molecular design* 26.7 (2012), pp. 801–804. DOI: 10.1007/s10822-012-9577-7.

[War13] Colin Ware. *Information visualization: Perception for design*. Morgan Kaufmann, 2013. ISBN: 9780123814654.

[Wea05] C. Weaver. "Visualizing coordination in situ". In: *InfoVis 05*. Ed. by John Stasko and Matt Ward. IEEE, 2005, pp. 165–172. ISBN: 0-7803-9464-x. DOI: 10.1109/infvis.2005.1532143.

[Weba] Website. *BirdEye*. https://code.google.com/archive/p/birdeye/. Last Accessed: 06-11-2024.

[Webb] Website. *GraphViz*. http://www.graphviz.org/. Last Accessed: 06-11-2024.

[Webc] Website. *Mathworks*. https://www.mathworks.com/. Last Accessed: 06-11-2024.

[Webd] Website. *Pajek*. http://mrvar.fdv.uni-lj.si/pajek/. Last Accessed: 06-11-2024.

[Webe] Website. *Rapidiner*. https://www.ibm.com/de-de/products/spss-statistics. Last Accessed: 06-11-2024.

[Webf] Website. *Rapidiner*. https://rapidminer.com/. Last Accessed: 06-11-2024.

[Webg] Website. *Sas*. https://www.sas.com/en_us/software/visual-analytics.html. Last Accessed: 06-11-2024.

[Webh] Website. *ScatterVis*. https://vca.informatik.uni-rostock.de/~ct/software/ScatterVis/. Last Accessed: 06-11-2024.

[Webi] Website. *Weka*. https://www.cs.waikato.ac.nz/ml/weka/. Last Accessed: 06-11-2024.

[Webj] Website. *Wolfram*. https://www.wolfram.com/. Last Accessed: 06-11-2024.

[Wen19] John Edward Wenskovitch Jr. "Dimension Reduction and Clustering for Interactive Visual Analytics". PhD thesis. Virginia Tech, 2019. DOI: 10.1109/TVCG.2022.3209423.

[WN13] W. Javed and N. Elmqvist. "ExPlates: Spatializing Interactive Analysis to Scaffold Visual Exploration". In: *Computer Graphics Forum* (2013), pp. 441–450. DOI: 10.1111/cgf.12131.

[Wol94] Pree Wolfgang. "Design patterns for object-oriented software development". In: *Reading Mass* 15 (1994).

[Won+03] Pak Chung Wong, Harlan Foote, Dan Adams, Wendy Cowley, and Jim Thomas. "Dynamic visualization of transient data streams". In: *Symposium on Information Visualization*. IEEE, 2003, pp. 97–104. DOI: 10.1109/infvis.2003.1249014.

[Won+16] Kanit Wongsuphasawat, Dominik Moritz, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. "Voyager: Exploratory Analysis via Faceted Browsing of Visualization Recommendations". In: *Transactions on Visualization and Computer Graphics* 22.1 (2016), pp. 649–658. DOI: 10.1109/tvcg.2015.2467191.

[Won+17] Kanit Wongsuphasawat, Zening Qu, Dominik Moritz, Riley Chang, Felix Ouk, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. "Voyager 2". In: *Explore, innovate, inspire*. Ed. by Gloria Mark, Susan Fussell, Cliff Lampe, m.c schraefel m.c, Juan Pablo Hourcade, Caroline Appert, and Daniel Wigdor. Association for Computing Machinery Inc. (ACM), 2017, pp. 2648–2659. ISBN: 978-1-4503-4655-9. DOI: 10.1145/3025453.3025768.

[WT04] Pak Chung Wong and Jim Thomas. "Visual analytics". In: *Computer Graphics and Applications* 24.5 (2004), pp. 20–21. ISSN: 02721716. DOI: 10.1109/mcg.2004.39.

[Wul21] Conrad Wulf. "Interaktive Vereinheitlichung von unterschiedlichen Parametern durch eine KI-basierte Ontologie". Master Thesis. University of Rostock, 2021.

[WYH09] Guiling Wang, Shaohua Yang, and Yanbo Han. "Mashroom: end-user mashup programming using nested tables". In: *Proceedings of the 18th international conference on World wide web*. 2009, pp. 861–870.

[XCZ15] Hanfa Xing, Jun Chen, and Xiaoguang Zhou. "A Geoweb-Based Tagging System for Borderlands Data Acquisition". In: *ISPRS International Journal of Geo-Information* 4.3 (2015), pp. 1530–1548. ISSN: 2220-9964. DOI: 10.3390/ijgi4031530.

[Yan+08] Fan Yang, Nitin Gupta, Chavdar Botev, Elizabeth F. Churchill, George Levchenko, and Jayavel Shanmugasundaram. "WYSIWYG development of data driven web applications". In: *Proceedings of the VLDB Endowment* 1.1 (2008), pp. 163–175. ISSN: 21508097.

[YS17] Bowen Yu and Claudio T. Silva. "VisFlow - Web-based Visualization Framework for Tabular Data with a Subset Flow Model". In: *Transactions on Visualization and Computer Graphics* 23.1 (2017), pp. 251–260. DOI: 10.1109/tvcg.2016.2598497.

[Yu+08] Qi Yu, Xumin Liu, Athman Bouguettaya, and Brahim Medjahed. "Deploying and managing Web services: issues, solutions, and directions". In: *The VLDB Journal* 17.3 (2008), pp. 537–572. ISSN: 10668888. DOI: 10.1007/s00778-006-0020-3.

[ZH13] Jinson Zhang and Mao Lin Huang. "5Ws Model for Big Data Analysis and Visualization". In: *Proceedings of the 16th International Conference on Computational Science and Engineering*. IEEE, 2013, pp. 1021–1028. DOI: 10.1109/cse.2013.149.

[Zha+12] Leishi Zhang, Andreas Stoffel, Michael Behrisch, Sebastian Mittelstadt, Tobias Schreck, Rene Pompl, Stefan Weber, Holger Last, and Daniel Keim. "Visual analytics for the big data era — A comparative review of state-of-the-art commercial systems". In: *Conference on Visual Analytics Science and Technology (VAST), 2012*. Ed. by Giuseppe Santucci. IEEE, 2012, pp. 173–182. ISBN: 978-1-4673-4753-2. DOI: 10.1109/vast.2012.6400554.

[Zie19] Michael Zielinski. "Koordiniertes Workflow Management mit Hilfe von Brushing & Linking Mechanismen". Bachelor Thesis. University of Rostock, 2019.

[Zie20] Michael Zielinski. "Interaktionsbasierter Datenaustausch zwischen unabhängigen Visual Analytics Werkzeugen". Master Thesis. University of Rostock, 2020.

[Zim+17] Michael Zimmermann, Felix W. Baumann, Michael Falkenthal, Frank Leymann, and Ulrich Odefey. "Automating the Provisioning and Integration of Analytics Tools with Data Resources in Industrial Environments Using OpenTOSCA". In: *2017 IEEE 21st International Enterprise Distributed Object Computing Workshop (EDOCW)*. IEEE, 2017, pp. 3–7. DOI: 10.1109/edocw.2017.10.

[Zur04] Michael Zur Muehlen. "Organizational Management in Workflow Applications – Issues and Perspectives". In: *Information Technology and Management* 5.3/4 (2004), pp. 271–291. ISSN: 1573-7667. DOI: 10.1023/B:ITEM.0000031582.55219.2b.

List of Acronym and Symbols

Index of Definitions

Flink Open-source stream- and batch-processing framework by the Apache Software Foundation
GNU Open-source operating system from 1984 by Richard Stallman
MatLab Proprietary programming language for numerical computing by MathWorks
NumPy Library for the Python programming language
SAS Programming language for Statistical Analysis by Anthony James Barr
R Programming language for Statistical Computing and Data Visualization by Ross Ihaka and Robert Gentleman
Scikit-learn Open-source machine learning library for the Python programming lanaguage
SciLab Open-source programming language for numerical computing
Tomcat Open-source implementation of Jakarta Servlet to provide Java web application servers by the Apache Software Foundation

Index of Abbreviations

ACID Atomicity, Consistency, Isolation, and Durability transaction properties
AnyProc Analytical Process Configurator
API Application Programming Interface
BPE Business Process Engineering
BPM Business Process Modeling
BRETAM Breakthrough, Replication, Empiricism, Theory, Automation, Maturity Learning model
CSS Cascading Style Sheets
DBMS Database Management System
DFG Deutsche Forschungsgemeinschaft
DSRM Data State Reference Model
DOM Document Object Model
ECMS Enterprise Content Management System
ERP Enterprise Resource Planning
ESB Enterprise Service Bus
GATE General Architecture for Text Engineering
GIMP GNU Image Manipulation Program

HCI Human Computer Interaction
HTML Hyper Text Markup Language
HTTP Hypertext Transfer Protocol
IBM International Business Machines
IEEE Institute of Electrical and Electronics Engineers
IVTK Information Visualization Toolkit
IT Information Technology
ITK Insight Toolkit
JSON JavaScript Object Notation
KNIME Konstanz Information Miner
MCV Multiple Coordinated Views
MES Manufacturing Execution System
MVC Model-View-Controller
OASIS Organization for the Advancement of Structured Information Standards
OLAP Online Analytical Processing
OpenCV Open Computer Vision Library
ORC Optical Character Recognition
PDA Production Data Acquisition
PDM Product Data Management
RDBMS Relational Database Management System
RDF Resource Description Framework
REST Representational State Transfer
RSS Rich Site Summary
SAP Systems, Applications and Products in Data Processing
SAS Statistical Analysis System
SOA Service-Oriented Architecture
SOAP Simple Object Access Protocol
SPSS Statistical Package for the Social Sciences
SQL Structured Query Language
TOSCA Topology and Orchestration Specification for Cloud Applications
UI User Interface
UIMA Unstructured Information Management Architecture
UnIVA Univeral Interface for Visual Analytics
UCSC University of California Santa Cruz
URI Uniform Resource Identifier
VAT Visual Analytics Tool
VTK Visualization Toolkit
Web World Wide Web (Internet)
WEKA Waikato Environment for Knowledge Analysis
WSDL Web Services Description Language
XML Extensible Markup Language
YAML Yet Another Markup Language

List of Figures

List of Tables

Danksagung

Das Thema dieser Dissertation entstand während der Arbeit am DFG-geförderten Forschungsprojekt UnIVA und wurde am Lehrstuhl für multimediale Kommunikation an der Universität Rostock entwickelt. Im Rahmen des Projektes konnte ich insgesamt 4 Jahre an den aktuellen Forschungsfragen arbeiten und die Ergebnisse in zahlreichen internationalen Werken veröffentlichen. Bevor ich einige Personen direkt anspreche, möchte ich folglich allgemein allen Fördergebern und Beteiligten für die Chance und die gute Zusammenarbeit in diesem Projekt danken.

Mein oberster Dank gilt zuallererst meinem Doktorvater Prof. Bodo Urban. Er hat mich während der Projektlaufzeit tatkräftig unterstützt und war selbst in schlechten Phasen so gut es ihm möglich war für mich erreichbar. Leider konnte Prof. Urban die Fertigstellung dieser Dissertation nicht miterleben, da er am 10.03.2022 an den Folgen seiner schweren Erkrankung verstorben ist. Durch seine sympatische und offene Art und seine Rollen als Lehrstuhlinhaber und Institutsleiter war Prof. Urban ein zentraler Anlaufpunkt für Feedback und half mir eine Expertennetzwerk aus zahlreichen Kolleg*innen der Universität Rostock und des Fraunhofer Instituts für graphische Datenverarbeitung (IGD) aufzubauen. Ich bin daher ebenso dankbar, dass sich die neue Institutsleitung des Fraunhofer IGDs Prof. Uwe von Lukas dazu bereit erklärt hat, die Betreuung in der finalen Phase dieser Arbeit zu übernehmen.

Ein besonderer Dank geht weiterhin an meine direkten Projektpartner*innen Prof. Heidrun Schuman und Dr. Martin Röhlig. Zusammen konnten wir in dem Projekt UnIVA große Erfolge im Bereich der Visual Analytics Forschung erreichen, dessen Ergebnisse diese Dissertation prägten und formten. Dies gilt auch für unsere externen Kooperationsparter*innen Prof. Hans-Jörg Schulz und Dr. Marius Hogräfer von der Universität Aarhus. Beide sind hochgradige Experten in ihrem Bereich und ich hoffe, dass wir auch in zukünftigen Projekten wieder zusammenarbeiten können.

Durch meine Nähe zum Fraunhofer IGD Rostock konnte ich im Rahmen des Visual Computing Research and Innovation Center (VCRIC) weitere wissenschaftliche Mitarbeiter*innen aus der praxisnahen Anwendungsentwicklung kennenlernen. Dabei gilt mein Dank vor allem den Teammitgliedern aus der Abteilung für Visual Assistence Technologies (VAT) vertreten durch Dr. Mario Aehnelt, welcher mir bei meiner persönlichen Entwicklung half und mir weiteren Erfolgschancen im Bereich der visuellen Assistenzforschung ermöglichte. Besonders möchte ich hier auch nochmal meinen direkten Büropartner Holger Diener nennen, welche mir zahlreichen Hinweise und Ratschläge bei der Bearbeitung des Themas geben konnte.

Mein Dank gilt nicht zuletzt meiner Frau Vivian Nonnemann, meinen Eltern Markus und Grit Nonnemann und meiner restlichen Familie und meinen Freunden, die mich während dieser anstregenden aber dennoch erfüllenden Promotionsphase unterstützt und stets weiter motiviert haben.

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Promotionsarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die von mir angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Werken wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

I hereby declare on oath that this submitted thesis was written by myself without any external assistance other than the sources and resources I have indicated. Passages, which have been taken out of third party publications of all means, either in whole or in part, in words or ideas, have been clearly marked as quotations in the referring passage. This work as a whole piece has never been part of any other examination nor it has been published before the date of defence.

Datum: 11. Juli 2025

Lars Nonnemann