A Characterization of Data Exchange between Visual Analytics Tools

Lars Nonnemann*, Heidrun Schumann*, Bodo Urban*, Mario Aehnelt† and Hans-Jörg Schulz‡
*Institute of Visual and Analytic Computing, University of Rostock, Rostock, Germany
Email: lars.nonnemann@uni-rostock.de, heidrun.schumann@uni-rostock.de, bodo.urban@uni-rostock.de
†Competence Center for Visual Assistance Technologies, Fraunhofer Institute for Computer Graphics Research, Rostock, Germany
Email: mario.aehnelt@igd-r.fraunhofer.de
‡Department of Computer Science, Aarhus University, Aarhus, Denmark
Email: hjschulz@cs.au.dk

***Abstract*—Over the past years, the visualization of large and complex data sets brought up various Visual Analytics (VA) tools in order to solve domain-specific tasks. These VA tools are typically implemented as individual software components in data-flow-oriented models, meaning that data is transferred from one component to the next. While most VA frameworks rely on a monolithic architecture with features for the integration of specialized analysis methods, we consider a loose coupling of independent applications, where autonomous VA tools are used in predefined analysis sequences. To this end, we provide a characterization of the data exchange process among individual VA tools in the form of a taxonomy. This taxonomy can be used as a checklist to identify characteristics and improve the data flow of one's own multi-tool VA setup. For this purpose, we conducted a systematic investigation of the individual aspects of data exchange that are commonly found across different usage scenarios. We apply our taxonomy to three existing multi-tool frameworks, the open-source library ReVize, the toolchain editor AnyProc, and the visualization and monitoring framework Plant@Hand3D.**

***Index Terms*—Data Analysis, Visual Analytics, Data Exchange**

1. Introduction

Visual Analytics (VA) tools are available in a variety of proprietary or open-source software, from analysis libraries [1] to full-fledged frameworks [2]. A common abstraction to be found in these VA solutions is that of a *data-flow* modeling how data is exchanged between individual software components (often termed *operators*) of a VA framework. This conceptual understanding of data being funneled through a series of operators, each making its changes and then handing the results off to the next operator, was first captured in Chi's *Data State Reference Model* [3]. Heer and Agrawala later enshrined it in the *Operator* design pattern for visualization software [4], which stands at the core of many of today's VA solutions, such as KNIME [5] or VisFlow [6].

Today, specialized application domains require specific analysis methods that are often not included in the generic VA frameworks. Instead multiple, developed standalone tools are used within their respective application domain. This is the point where the conventional intra-tool data-flow model ends, and the jerry-rigging of some form of inter-tool data exchange via custom scripts starts. Consequently, the need for a more structured and reusable form of data exchange between independent applications is echoed throughout many application fields from biology [7] to geosciences [8].

Some approaches to exchange data across tools have been explored in the past. For example, North and Shneiderman's *Snap-Together* visualizations [9] use an underlying relational database system as a central broker among independent tools. Whereas Rogowitz and Matasci's *Meta-Data Mapper* [10] utilizes a service-based infrastructure to distribute data to multiple tools. Yet, these software architectural considerations are only one side of the coin. The other side being how the tools are actually used together: Are the tools used only once or repeatedly, subsequently or concurrently, symmetrically or asymmetrically, etc.? We strongly believe that these two sides of the same coin must match each other – i.e., the underlying architecture must be a good fit to enact the necessary prerequisites for data exchange for a given cross-tool VA usage scenario.

In this paper, we systematically investigate how data exchange between independent VA tools can be realized. By answering the five W-questions of *What kind of data is exchanged?*, *Why, Where, and When is it exchanged?*, and finally *Which type of access is granted to the tool?* we create a taxonomy that describes the most important characteristics of data exchange. Such a taxonomy can be advantageous in several ways. Kerracher and Kennedy [11] point out that taxonomies help us to specify the "space of possible" and make sense of what already exists. Moreover, they argue that taxonomies provide a common vocabulary that allows us to describe and compare different approaches. In particular, taxonomies can help systematize the design by abstracting from domain specifics and underlying technologies. With our taxonomy of data exchange between VA tools, we aim to

support the evaluation, comparison, and design of different exchange strategies. To substantiate these aspects, we apply our taxonomy to three demonstrating examples.

2. Related Work

Each software or hardware solution for VA has three layers that must fulfill the analyst's requirements: data management, analytics, and visualization [12]. These three layers cover separate research areas on their own, each coming with its own significant challenges [13]–[15]. As a result, research questions regarding these layers are often dealt with in isolation, developing solutions that work on one layer, but are ill-fitting for the others.

Recently, data management and analysis tools have begun to converge through the use of multiple technologies, including grids, cloud computing, and general-purpose graphics processing [12], [16]. Therefore, data management and visual analysis get involved in the creation of data-flow-oriented models. In 1995, Lee and Parks [17] introduced one of the most common data-flow models for VA systems, and it is still widely used in various visualization systems, including ConMan [18], AVS [19], SCIRun [20], and VTK-based systems such as Paraview [21], VisIt [22], and VisTrails [23]. These systems incorporate different data management, analysis, and visualization operators into an overall system, and implement the data exchange between these modules in a system-specific manner. However, when it comes to conducting data analysis using a number of independent VA tools, such system-specific solutions are no longer applicable.

The idea of analyzing data across a set of multiple independent tools in a sequential form is well established and captured in the concept of *analysis pipelines*. In contrast, more flexible analysis scenarios of using different tools repeatedly in a back-and-forth manner or even concurrently in a side-by-side manner have only recently been introduced [24], [25]. The main idea behind them is to generalize the unidirectional linear analysis pipeline into a more encompassing graph model, representing VA tools as its nodes, and different communication layers between VA tools as directed edges. The subsequent use of VA tools charts a path through that graph, which we call *toolchain* [26]. These toolchains describe a flexible, loose coupling of tools that follows a specific analysis process, rather than their integration into an "all-encompassing" VA system. However, it is hardly investigated in more detail how the data can be exchanged between such individual VA tools. As for most visualization systems, data exchange is usually expected to work, but far away from a systematic treatment. This might be due to the overlap with the area of data management.

Data management is a longstanding research topic of computer science with the goal to ensure data consistency, avoid duplication and handle data transactions in a formal way. Most traditional database management systems rely on relational database models to exchange and integrate data by exploiting a highly optimized and standardized data access interface through the SQL query language. Over the past years, modern solutions such as data warehousing, OLAP (On-Line Analytical Processing), and data mining have been applied in this area more often, as these techniques are intended to support visualizations, strategic analyses, and decision processes. Obstacles for data management stem from the fact that data sets are often very large and growing incrementally, while their sources are heterogeneous, autonomous, and typically distributed. It becomes increasingly complex to provide unified and transparent access to large volumes of data in an organized way.

Thus, the management and transformation of data are an essential part of VA research, as described by Thomas and Cook [27], Keim et al. [16], and Fekete [12]. Nevertheless, we found that data exchange between independent VA tools is usually not discussed. This might be due to the immense wealth of modern information systems in terms of acquisition, computation, and storage, which are usually no primary concerns for visualization research. In this paper, we aim at bridging this gap by introducing a taxonomy to characterize the data exchange among individual VA tools.

3. A Taxonomy of Data Exchange between VA tools

Our taxonomy is meant to provide a classification of data exchange between independent VA tools in order to capture the range of possibilities by detailing their individual characteristics. Our classification approach follows the *Five W's Model* [28] that asks for different aspects of a concept through five questions of *What?*, *Why?*, *Where?*, *When?*, and *Who?*. For our case, we adjusted the *Who?* into a *Which?*, as we are talking about the communication between VA tools instead of users. This leaves us with five characterizing questions shown in Figure 1.

In the following, we look at these questions in detail and identify common answers to each – i.e., the resulting characteristics. For each of these characteristics, we then discuss their implications for the data-flow-oriented analysis between multiple VA tools in three demonstrating examples summarized in Section 4.

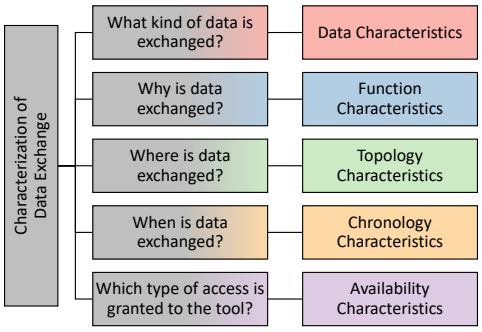

Figure 1. Our five characterizing questions for data exchange between independent VA tools

3.1. Data Characteristics

Data is a versatile term usually used for a certain quantity of information ranging from a huge number of values in a data source to a few parameters synchronized between two applications. Starting from this basic idea about data, we investigate the diversity of this term by answering the following questions: *What type of information can be extracted from a data source?*, *What type of information is commonly used by tools?*, and *What partition of information should be exchanged?*.

3.1.1. Informational Contents. If we talk about data for visual analysis, we usually refer to it as some sort of variables that hold information about the usage scenario such as n-dimensional vector fields, 3D geometry, or geospatial sensor-data. However, this is not the only type of informational content that is used for analytical tasks.

A *Data set* is a collection of values, items, or facts, which are described by primitives (for example numbers, strings, or vectors) based on predefined value scales. 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.

A *Data descriptor*, on the other side, encapsulates basic information about the data set such as provenance information, storage schemata, uncertainties, or general metadata [29]. This type of information can be used by the overarching system during the subsequent analysis to automatically recognize possible data transformations, categorize similar VA tools, or adequately parametrize views and highlight or exclude data items. Examples of such metadata are semantic information through tags or keywords as in HTML or CSS elements, or editorial information such as date, time, and authorship in stored documents.

3.1.2. 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 therefore a challenge, which is discussed in various research papers [13], [30] and patent applications [31], [32]. In order to understand the fundamental differences of data from an abstract view, it is therefore common to categorize data on their structuredness.

Structured Data refers to organized information that follows a predefined data model (also called schema) [33]. These types of data formats are typically used in relational databases management systems (RDBMS), where fields store attributes in different columns of a table [34]. Common examples for structured data are reports, logs, tags, or objects in large repositories (e.g. payroll, inventory, or account management).

Unstructured Data is essentially the opposite as it includes data with no pre-defined schema [33]. Unstructured data originates from machine- or human-generated information and is categorized into two common types [34]: *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 includes predefined structure elements such as tags or other markers and free-form unstructured components to separate semantic elements and enforce hierarchies of records and fields within the corresponding information [35], [36]. Therefore, it is also known as schemaless or self-describing structure. Common examples for semi-structured data formats are usually based on the extensible Markup-Language (XML) or the Javascript object notation (JSON).

3.1.3. Quantity of Exchanged Data. The previous sections showed that the information within a data source can be diverse. 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. We describe this quantity using three different characteristics.

A *Full Data Exchange* is the simplest solution for the information transfer, as the entire data set is exchanged as a whole. A common example are snapshots, which can store the state of a system at a certain time [37]. This is especially useful for a coherent analysis experience forward and backward along toolchains.

A *Segmented Data Exchange* is used to enhance the performance of toolchains. Therefore, data is split into segments over time to provide a fast and efficient information processing. This is especially important for the progressive refinement of visualizations [38]. The way in which segmentation is performed depends strongly on the data type. For example, while graph data is divided topologically, whereas time data is partitioned into sensible intervals.

A *Delta Data Exchange* is carried out by exchanging only a the changed part of the data source that is selected based on the urgency or priority of operations. Examples for this are small selections of groups or single parameter manipulations in one VA tool that result in an immediate update of the visual representation in another VA tool.

3.2. Function Characteristics

After knowing about 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?*

The *Import* and *Export* of data is the most fundamental operation in our data exchange process, as we assume that VA tools consume and produce information from data sources. The full data set is thereby exchanged in order to be used by the corresponding VA tool.

It is inevitable that information in a data source will be changed during the analysis. This *Modification* of data is commonly performed by saving the progress of one VA tool 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* of 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. An example of this would be streaming services, where frequent updates are performed by sending the new generated data over the system's infrastructure to the corresponding VA tool [39], [40].

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 [41], where selections in one tool trigger a highlighting in another tool. This process is usually realized through a bidirectional delta data exchange.

3.3. Topology Characteristics

So far, we considered data exchange as a communication process. However, in order to perform the communication, there needs to be some sort of medium. Thus, we need to cover the questions *Where is the data exchanged?*, *Where does the data come from?*, and *Where is the data going?*

3.3.1. Infrastructure. There are multiple software infrastructures for the data exchange between different tools. However, all of these infrastructures can be divided in two fundamental architectures.

A *Centralized Architecture* is a single software solution that exchanges data between multiple VA tools through a uniform platform. The typical way of pursuing this is by using a relational database that acts as model in a model-view-controller mechanism. Examples for such a centralized data storage are systems like Snap-together [9] or EdiFlow [42]. Another possibility for a centralized architecture is the employment of a service bus to broker messages among tools. An example for such a service-oriented architectures is the Metadata Mapper [10].

A *Decentralized Architecture* provides data exchange through a variety of different mechanisms without a uniform platform. Therefore, custom connectors are applied in order to access content from websites or shared network data. Common techniques for this are web mining and extensive browsing as in Intertwine [43], or mashup tools like Mashroom+ [44] or VisMashup [45].

3.3.2. Relations. Previous work describes VA tools as *black boxes* that can handle input and output information [26]. Following up on this, we define data sources to be either initial input or resulting output information of a VA tool that may be stored in a data format. Hence, with multiple tools, it is obvious that there are multiple possibilities for inputs and outputs. Therefore, we consider four key relations between data sources and VA tools.

The *One-to-One Relationship* provides each tool with exactly one data source. This is the simplest use case that could be used for the initialization of a VA tool or the transport of output data between two subsequently used VA tools.

The *One-to-Many Relationship* is applied when multiple data sources are used by a single VA tool. For this to work, data transformation such as merging between the data sets needs to be applied for all used data sources. This problem gets increasingly hard with a multitude of data formats.

The *Many-to-One Relationship* refers to one data source that is used by multiple tools. Clear communication between the independent VA tools about the chronological order is needed, as simultaneous access to information can lead to problems with multiple versions of the same data source.

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

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

For the *Unilateral data exchange*, data is transmitted exclusively from one VA tool which is the sender, to another VA tool, 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 that are used only once, without the need for specific parameter adjustments. Therefore, data is just passed through in one direction of the toolchain.

The *Bilateral data exchange*, on the other side, offers an open communication between VA tools in both directions. This corresponds to continuous analysis tasks with frequent parameter changes such as the parallel display or dynamic switches between VA tools. Example use cases are data exchanges between multi-functional tools that offer the right parameters for a comparative or advanced visualization of information.

3.4. 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 [46]. Therefore, we need to answer the questions *When is the information exchange planned?* and *When is the exchange executed?*

3.4.1. Timeline. We define the timeline for the data exchange process between two VA tools as a planned execution of different data transfers. A data transfer is thereby the process of passing portions of the data from one VA tool to another. We define this process to be either synchronous or asynchronous.

The *Synchronous Data Transfer* is performed, if both VA tools are available for sending and/or receiving data at the same time. This requires an open communication channel that is not closed during the exchange process.

The *Asynchronous Data Transfer* allows a delayed data exchange, where both VA tools are available for sending

	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 2. Our taxonomy for the characterization of data exchange between independent tools. Each aspect is thereby included in one of the five categories.

and/or receiving data at different times. This communication is usually 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 VA tools used during an analytical task, since performance can be affected by the high number of different simultaneous preparation steps.

3.4.2. Order. While the timeline of the data exchange process is used to plan the execution of the data exchange, there are also different strategies for the status of the data during the execution order. Di Lorenzo et al. [47] mentions two strategies that are based on the invocation of said VA tools.

The *Pull Strategy* is based on frequent and repetitive requests that are initiated by the receiving VA tool – e.g., in a data-flow model, an idle operator would poll its predecessor(s) in the toolchain for new data to work on. The polling frequency should thereby set to be lower than the average update frequency of the data source itself in order to control the amount of performed operations.

The *Push Strategy*, on the other side, is initiated by the sending VA tool – e.g., in a data-flow model, an operator that has completed its computation would send off its results

to its successor(s) in the toolchain. If the successors are not known to the operator, a central broadcast to all operators may also be possible where the individual operators then decides itself whether to accept or decline the data.

3.5. Availability Characteristics

So far, we considered data exchange as a process by which a receiving VA tool obtains information from a sending VA tool over the system's infrastructure. However, in order to carry out this process, we need to cover if data is even available for the exchange process by answering the questions of: *Which data can be accessed?* and *For which time period is the access granted?*

3.5.1. Restriction. Data can be obtained in many different ways, whether it is through a file system or databases, by adding data values as URL tokens or as inter-process communication. However, data may not always be available due to security restrictions.

In the case of *Full access* to data sources, information can be freely exchanged between tools over the system infrastructure.

However, sometimes *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.

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 [48], [49] 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 for the opposite direction of extracting information from an application's UI. If no structured information about the screen contents are available, this can be done via OCR and image processing [50]. Yet if such information is available – e.g., the Document Object Model (DOM) of a website – an extractor or wrapper can be used to find and obtain the relevant information [51]. Another way of information retrieval is the parsing of rendered user interfaces [48] 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 [52], and the Gaggle Tool Creator [53] or SideCache [54] and SideKick [55], which are used in the biomedical domain.

3.5.2. 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 VA tool in order to keep data available at any later point in time. The data is thereby only modified by the VA tool, which requested the access until all needed operations are finished. This ensures a consistency of the selected data within a VA tool. However, this behavior will lead to conflicts, if the data is shared between multiple VA tools.

Transient Access is granted to a VA tool for the duration of the exchange process itself. An example for this would be a unidirectional analysis process with a sequential use of multiple VA tools one after another. In this scenario, it is not necessary to keep access to a data source, as the analysis tasks enforces a subsequent execution of VA tools.

In summary, we have introduced a taxonomy with five categories to classify the data exchange between independent VA tools. The categories capture *what* kind of data is *why, where, and when* exchanged, and *which* type of access is granted for the tool. Each category describes particular aspects of the data exchange independent of the underlying technology (see Figure 2).

4. Demonstrating Examples

In this section, we showcase how our taxonomy can be used to describe the data exchange in existing multi-tool VA setups, as well as its ability to pinpoint those data exchange characteristics that require changes if new features are to be realized. To that end, we discuss three visual analytics frameworks: the open-source library ReVize, the toolchain editor AnyProc, and the visualization and monitoring framework Plant@Hand3D.

4.1. ReVize

Our first example is the open-source library ReVize [56] that can be used to add data exchange capabilities between different web tools using the Vega-Lite visualization grammar [57]. The ReVize framework makes use of the fact that Vega-Lite is able to declaratively describe certain aspects of the visualization toolchains from preprocessing the data to its mapping onto visual elements. This way, changes to any part of the visualization are reflected in the same visualization description, which simply gets updated and amended accordingly.

4.1.1. Classification. *What kind of data is exchanged?*

ReVize exchanges data sets without metadata in full. The data is exchanged as JSON and thus semi-structured.

Why is data exchanged?

ReVize specifically supports the import and export of visualizations to be used or modified in otherwise closed web tools that were originally not designed to handle Vega-Lite.

Where is data exchanged?

ReVize makes no assumptions about the infrastructure, relationships, or directionality of the data exchange.

When is data exchanged?

ReVize makes no assumptions about the timeline or order of the data exchange.

Which type of access is granted to the tool?

ReVize requires full and persistent access to the Vega-Lite description defining the visualization. Even if a VA

Data Characteristics	*Content*	*Formats*	*Quantity*
	Data Sets without Metadata	Semi-structured Data	Full Data Exchange
Function Characteristics	*Operations*		
	Import, Export and Modification		
Topology Characteristics	*Infrastructure*	*Relations*	*Directionality*
Chronology Characteristics	*Timeline*	*Order*	
Availability Characteristics	*Restriction*	*Retention*	
	Handles full access	Persistent access required	

Figure 3. Classification of data exchange using ReVize

tool merely changes some data values in the data object of a Vega-Lite specification, ReVize needs to parse this object in full and re-insert the changes afterwards.

4.1.2. Discussion. As a library, ReVize has one very specific goal: to handle the data input/output of Vega-Lite based visualization descriptions. Hence, it does not provide or specify any concrete means of actually getting the data from one VA tool to the next – see Figure 3. This way, one can even use the manual exchange of data via the clipboard as a lightweight possibility to couple VA tools via ReVize. Its versatility of integrating with any such means of data exchange makes ReVize a good fit for very heterogeneous ensembles of web-based VA tools in which no single mode of data transfer can be established.

One starting point for future extension of ReVize is to replace this approach with an automated one that allows for versioning of the exchanged Vega-Lite specifications, and thus for a cross-tool undo mechanism. Revisiting Figure 3 for this scenario points to the following changes:

Function Characteristics – Operations: The exchange mechanism needs to support *update* and *synchronization* operations, so as to inform VA tools about new versions of data sets and about roll-backs to an older version.

Topology Characteristics – Infrastructure: A *centralized architecture* will be necessary for the versioning. E.g., a Git server could be used to provide this functionality.

Chronology Characteristics – Timeline: The whole idea of versioning implies a synchronous data transfer to ensure that data updates (new version, or undo to an older version) are taken into account as they happen, so that all VA tools work on the current version of the data.

4.2. AnyProc

Our second example is a prototype application for the configuration and execution of toolchains – i.e. AnyProc (analytical Process Constructor). It provides a visual editor that allows us to couple data sources and VA tools as nodes of a directed graph. The created toolchains can further be saved to be executed in a step-by-step manner. The data exchange between the linked VA tools is based either on loading data from a data source or transferring data from a previously executed VA tool to the next one. A first version of AnyProc is available under Public License as a download on the Website of the Visual Computing Research and Innovation Center [58].

4.2.1. Classification. *What kind of data is exchanged?*

AnyProc makes no assumptions about the content or formats of a data source, but instead uses references to the connected VA tools to import the connected data through independent commands. Data is exchanged as a whole.

Why is data exchanged?

The core idea of AnyProc surrounds the import and export of data sets between independent VA tools. Thus, no other operations are considered for the visual analysis tool as modification is only done within each independent VA tool and further exported as a new data source.

Where is data exchanged?

Information in AnyProc is exchanged without any central infrastructure by linking data source and tool nodes through the graph model. AnyProc makes no assumptions about the relations or direction of data exchange between VA tools, since both depend on the output of the previous VA tools.

When is data exchanged?

The data exchange process is manually started by the user through a small navigation window to synchronously push information from one VA tools to the next.

Which type of access is granted to the tool?

Our current prototype can handle any type of local or remote information as long as there is a compatible VA tool that has persistent access to it.

4.2.2. Discussion. As a visual editor for the configuration of analytical processes, AnyProc provides the necessary flexibility for the loose coupling of independent VA tools. Therefore, it does not aim to specify data source structures or operations performed within each VA tools (see Figure 4).

However, the incompatibility of different data source is a major problem for the current prototype application. Therefore, concepts of data conversion should be considered in further development. Revisiting Figure 4 for this scenario points to the following changes:

Data Characteristics	*Content*	*Formats*	*Quantity*
			Full Data Exchange
Function Characteristics	*Operations*		
	Import and Export		
Topology Characteristics	*Infrastructure*	*Relations*	*Directionality*
	Decentralized exchange between tools		
Chronology Characteristics	*Timeline*	*Order*	
	Synchronous exchange	Push strategy	
Availability Characteristics	*Restriction*	*Retention*	
	Handles full access	Persistent access required	

Figure 4. Classification of data exchange in AnyProc

Data Characteristics – Contents & Formats: For the conversion of data sources, it is necessary to know whether the contents is enriched with *data descriptors* or if the tools are handling pure *data sets*. Furthermore the systems requires to know about the used formats for each VA tool to provide possible conversions for structured, semi-structured or unstructured data.

Function Characteristics – Operations: Conversion itself is a modification of data. This aspect needs to be included in the current data exchange characterization.

4.3. Plant@Hand3D

Our last example is Plant@Hand3D [59], a system for the visualization and real-time monitoring of industrial manufacturing processes. The three-dimensional model of the factory site acts as a digital twin, which enables fast switches between several existing VA tools embedded through integrated application windows.

4.3.1. Classification. *What kind of data is exchanged?*

Plant@Hand3D exchanges data from enterprise databases as well as raw sensors data, which are available as either structured or semi-structured input formats for the corresponding VA tools. The data is thereby exchanged as a whole or through delta data exchange.

Why is data exchanged?

Plant@Hand3D aims to model and visually encode domain environments by providing an interface that enables import, modification, updates, and synchronization between multiple VA tools.

Where is data exchanged?

Plant@Hand3D uses a centralized service bus that offers a bidirectional communication between multiple VA tools through One-to-One relationships.

When is data exchanged?

The exchange process is initiated for a synchronous pull strategy each time a new VA tool is opened up by interacting with the three dimensional model.

Which type of access is granted to the tool?

Its infrastructure offers a wide accessibility for databases, web services, and local file systems. However, data is only used if it is supported by the service architecture in terms of content, format, and restriction of data.

4.3.2. Discussion. As a comprehensive visual analysis tool, Plant@Hand3D provides the user with fully specified data exchange capabilities to model real-world environments in a digital twin of industrial factories. Therefore, all characteristics for data exchange are described by the technical infrastructure (see Figure 5). Its organized infrastructure and visually compelling interface makes Plant@Hand3D a stable foundation for various usage scenarios. A first step in this direction is an inherited application for the medical domain called Health@Hand [60], which provides a module for the storage of predefined toolchains.

However, the alteration of usage scenarios results sometimes in an unexpected amount of data to be processed.

Data Characteristics	*Content*	*Formats*	*Quantity*
	Data Sets without Metadata	Structured or Semi-structured Data	Full and Delta Data Exchange
Function Characteristics	*Operations*		
	Import, Modification, Updates and Synchronization		
Topology Characteristics	*Infrastructure*	*Relations*	*Directionality*
	Centralized exchange through a service bus	One-to-One and One-to-Many Relationships	Bidirectional
Chronology Characteristics	*Timeline*	*Order*	
	Synchronous exchange	Pull strategy	
Availability Characteristics	*Restriction*	*Retention*	
	Handles full access	Persistent access required	

Figure 5. Classification of data exchange in Plant@Hand3D

Therefore Plant@Hand3D would benefit from improvements in performance and interaction feedback. One approach to achieve this could be the increased use of progressive analytics. Revisiting Figure 5 for this scenario points to the following changes:

Data Characteristics – Quantity: To reduce the amount of processed information through progressive analytics, it is necessary, that Plant@Hand3D establishes a *segmented data exchange*.

Chronology Characteristics – Timeline & Order: Beside the reduction for the spatial reduction of information, it is also required to make use of the time-oriented distribution of content. Therefore, a push strategy should be applied for asynchronous updates in different parts of the system whenever changes appear.

5. Conclusion

In this paper, we presented a taxonomy for the classification of data exchange between independent VA tools. To this end, we performed a systematic investigation for different aspects of the data exchange process based on a refined definition of the *Five W's Model* [28]. We found *data, function, topology, chronology*, and *availability characteristics* to classify data exchange between multiple VA tools. Our resulting taxonomy (see Figure 2) is therefore a generalized solution, that describes the problem of data exchange on an abstract layer without considering the underlying technology. Therefore it can be applied to specific implementations in order to classify characteristic aspects and find criteria for the evaluation, comparison, and design of visual analysis tools. We demonstrated the feasibility of this approach on three existing visual analysis tools, each offering a different way for the loose coupling of VA tools. We discussed the found "space of possible" for each of these systems to provide 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

The comprehensive digital twin Plant@Hand3D [59] offered a full definition of data exchange through our taxonomy, while the prototype application AnyProc [58] and

the description library ReVize [56] had some missing values for different aspects of our characterization. However, 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: While Plant@Hand3D offers a specialized model for data exchange that is strictly bound to its implementation architecture, ReVize and AnyProc offer a flexible approach that leaves some technical challenges unanswered. The discussion for each of these systems underlines that our taxonomy can be used to pinpoint those aspects of data exchange that need to be altered for providing further extensions, like version control, data transformation, or progressive analytics.

In the future, we envision that our taxonomy can be enhanced from research communities with different perspectives to provide a rich set of characteristics for data exchange processes. Furthermore, we want to investigate the dependencies of VA tools and specific usage scenarios to define rules for the construction and identify possibilities for the combination or reduction of toolchains. We think that this approach could support the user by performing tedious tasks automatically or semi-automatically so that the domain expert can focus on analyzing information at different layers of abstraction.

Acknowledgments

We thank the anonymous reviewers for their thoughtful comments, as well as the German Research Foundation (DFG) for financial support of this research within the project UnIVA.

References

[1] E. Ventocilla and M. Riveiro, "Visual Analytics Solutions as 'off-the-Shelf' Libraries," in *Proc. of IV*. IEEE, 2017, pp. 281–287.

[2] M. Behrisch, D. Streeb, F. Stoffel, D. Seebacher, B. Matejek, S. H. Weber, S. Mittelstädt, H. Pfister, and D. Keim, "Commercial Visual Analytics Systems–Advances in the Big Data Analytics Field," *Transactions on Visualization and Computer Graphics*, vol. 25, no. 10, pp. 3011–3031, 2019.

[3] E. H.-H. Chi and J. T. Riedl, "An operator interaction framework for visualization systems," in *Proc. of IV*. IEEE, 1998, pp. 63–70.

[4] J. Heer and M. Agrawala, "Software Design Patterns for Information Visualization," *Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 853–860, 2006.

[5] M. R. Berthold, N. Cebron, F. Dill, T. R. Gabriel, T. Kötter, T. Meinl, P. Ohl, K. Thiel, and B. Wiswedel, "KNIME – The Konstanz Information Miner: Version 2.0 and Beyond," *SIGKDD Explorations Newsletter*, vol. 11, no. 1, pp. 26–31, 2009.

[6] B. Yu and C. T. Silva, "VisFlow - Web-based Visualization Framework for Tabular Data with a Subset Flow Model," *Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 251–260, 2017.

[7] J. C. Bare and N. S. Baliga, "Architecture for interoperable software in biology," *Briefings in Bioinformatics*, vol. 15, no. 4, pp. 626–636, 2012.

[8] P. Baumann, P. Mazzetti, J. Ungar, R. Barbera, D. Barboni, A. Beccati, L. Bigagli, E. Boldrini, R. Bruno, A. Calanducci, P. Campalani, O. Clements, A. Dumitru, M. Grant, P. Herzig, G. Kakaletris, J. Laxton, P. Koltsida, K. Lipskoch, A. R. Mahdiraji, S. Mantovani, V. Merticariu, A. Messina, D. Misev, S. Natali, S. Nativi, J. Oosthoek, M. Pappalardo, J. Passmore, A. P. Rossi, F. Rundo, M. Sen, V. Sorbera, D. Sullivan, M. Torrisi, L. Trovato, M. G. Veratelli, and S. Wagner, "Big Data Analytics for Earth Sciences: the EarthServer approach," *International Journal of Digital Earth*, vol. 9, no. 1, pp. 3–29, 2016.

[9] C. North and B. Shneiderman, "Snap-together Visualization: A User Interface for Coordinating Visualizations via Relational Schemata," in *Proc. of AVI*. ACM, 2000, pp. 128–135.

[10] B. E. Rogowitz and N. 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*, B. E. Rogowitz and T. N. Pappas, Eds. SPIE, 2011, vol. 7865, pp. 165–177.

[11] N. Kerracher and J. Kennedy, "Constructing and Evaluating Visualisation Task Classifications: Process and Considerations," *Computer Graphics Forum*, vol. 36, no. 3, pp. 47–59, 2017.

[12] J.-D. Fekete, "Visual Analytics Infrastructures: From Data Management to Exploration," *IEEE Computer*, vol. 46, no. 7, pp. 22–29, 2013.

[13] M. Magnani and D. Montesi, "A unified approach to structured, semistructured and unstructured data," University of Bologna, Department of Computer Science, Tech. Rep. 2004-9, 2004.

[14] D. Keim, G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, and G. Melançon, "Visual Analytics: Definition, Process, and Challenges," in *Information Visualization*, ser. Lecture Notes in Computer Science, A. Kerren, J. Stasko, J.-D. Fekete, and C. North, Eds. Springer, 2008, vol. 4950, pp. 154–175.

[15] A. Kadadi, R. Agrawal, C. Nyamful, and R. Atiq, "Challenges of data integration and interoperability in big data," in *Proc. of Big Data*. IEEE, 2014, pp. 38–40.

[16] D. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann, Eds., *Mastering the information age: Solving problems with visual analytics*. Eurographics Association, 2010.

[17] E. A. Lee and T. M. Parks, "Dataflow process networks," *Proc. of the IEEE*, vol. 83, no. 5, pp. 773–801, 1995.

[18] P. E. Haeberli, "ConMan: A visual programming language for interactive graphics," in *Proc. of SIGGRAPH*. ACM, 1988, pp. 103–111.

[19] 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," *Computer Graphics and Applications*, vol. 9, no. 4, pp. 30–42, 1989.

[20] S. G. Parker and C. R. Johnson, "SCIRun: A scientific programming environment for computational steering," in *Proc. of ICS*. ACM, 1995, p. 52.

[21] J. P. Ahrens, B. Geveci, and C. C. W. Law, "ParaView: An End-User Tool for Large-Data Visualization," in *The Visualization Handbook*, C. D. Hansen and C. R. Johnson, Eds. Elsevier, 2005, pp. 717–731.

[22] H. Childs, E. Brugger, B. Whitlock, J. Meredith, S. Ahern, D. Pugmire, K. Biagas, M. Miller, G. H. Weber, H. Krishnan *et al.*, "VisIt: An end-user tool for visualizing and analyzing very large data," in *High Performance Visualization: Enabling Extreme-Scale Scientific Insight*, E. W. Bethel, H. Childs, and C. Hansen, Eds. CRC Press, 2012, pp. 357–372.

[23] J. T. Morisette, C. S. Jarnevich, T. R. Holcombe, C. B. Talbert, D. Ignizio, M. K. Talbert, C. Silva, D. Koop, A. Swanson, and N. E. Young, "VisTrails SAHM: visualization and workflow management for species habitat modeling," *Ecography: Pattern and Diversity in Ecology*, vol. 36, no. 2, pp. 129–135, 2013.

[24] D. Gürdür, F. Asplund, J. El-khoury, F. Loiret, and M. Törngren, "Visual Analytics Towards Tool Interoperabilty: A Position Paper," in *Proc. of VISAPP*. SciTePress, 2016, pp. 139–145.

[25] H.-J. Schulz, M. Röhlig, L. Nonnemann, M. Aehnelt, H. Diener, B. Urban, and H. Schumann, "Lightweight Coordination of Multiple Independent Visual Analytics Tools," in *Proc. of VISAPP*. SciTePress, 2019, pp. 106–117.

[26] H.-J. Schulz, M. Röhlig, L. Nonnemann, M. Hogräfer, M. Aehnelt, B. Urban, and H. Schumann, "A Layered Approach to Lightweight Toolchaining in Visual Analytics," in *Computer Vision, Imaging and Computer Graphics Theory and Applications*, ser. Communications in Computer and Information Science, A. P. Cláudio, K. Bouatouch, and M. Chessa, Eds. Springer, 2020, pp. 313–337.

[27] J. J. Thomas and K. A. Cook, *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. IEEE, 2005.

[28] J. Zhang and M. L. Huang, "5Ws Model for Big Data Analysis and Visualization," in *Proc. of CSE*. IEEE, 2013, pp. 1021–1028.

[29] H.-J. Schulz, T. Nocke, M. Heitzler, and H. Schumann, "A systematic view on data descriptors for the visual analysis of tabular data," *Information Visualization*, vol. 16, no. 3, pp. 232–256, 2017.

[30] P. Lo Giudice, L. Musarella, G. Sofo, and D. Ursino, "An approach to extracting complex knowledge patterns among concepts belonging to structured, semi-structured and unstructured sources in a data lake," *Information Sciences*, vol. 478, pp. 606–626, 2019.

[31] D. R. Judd, B. Karsh, R. Subbaroyan, T. Toman, R. Lahiri, and P. Lok, "Apparatus and method for searching and retrieving structured, semi-structured and unstructured content," 2004, US Patent App. 10/439,338.

[32] F. 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," 2018, US Patent App. 15/387,070.

[33] M. Barbulescu, R. Grigoriu, I. Halcu, G. Neculoiu, V. C. Sandulescu, M. Marinescu, and V. Marinescu, "Integrating of structured, semi-structured and unstructured data in natural and build environmental engineering," in *Proc. of RoEduNet*. IEEE, 2013, pp. 1–4.

[34] K. Sambrekar, V. S. Rajpurohit, and J. Joshi, "A Proposed Technique for Conversion of Unstructured Agro-Data to Semi-Structured or Structured Data," in *Proc. of ICCUBEA*. IEEE, 2018, pp. 1–5.

[35] J. A. Sanchez, C. Proal, and F. Maldonao-Naude, "Supporting structured, semi-structured and unstructured data in digital libraries," in *Proc. of ENC*. IEEE, 2004, pp. 368–375.

[36] S. Abiteboul, P. Buneman, and D. Suciu, *Data on the Web: from relations to semistructured data and XML*. Morgan Kaufmann, 2000.

[37] M. Streit, H.-J. Schulz, A. Lex, D. Schmalstieg, and H. Schumann, "Model-driven design for the visual analysis of heterogeneous data," *Transactions on Visualization and Computer Graphics*, vol. 18, no. 6, pp. 998–1010, 2012.

[38] M. Angelini, G. Santucci, H. Schumann, and H.-J. Schulz, "A Review and Characterization of Progressive Visual Analytics," *Informatics*, vol. 5, no. 3, pp. 31:1–31:27, 2018.

[39] P. C. Wong, H. Foote, D. Adams, W. Cowley, and J. Thomas, "Dynamic visualization of transient data streams," in *Proc. of InfoVis*. IEEE, 2003, pp. 97–104.

[40] R. J. Crouser, L. Franklin, and K. Cook, "Rethinking Visual Analytics for Streaming Data Applications," *Internet Computing*, vol. 21, no. 4, pp. 72–76, 2017.

[41] M. Waldner, W. Puff, A. Lex, M. Streit, and D. Schmalstieg, "Visual links across applications," in *Proc. of GI*. Canadian Information Processing Society, 2010, pp. 129–136.

[42] V. Benzaken, J.-D. Fekete, P.-L. Hémery, W. Khemiri, and I. Manolescu, "EdiFlow: Data-intensive interactive workflows for visual analytics," in *Proc. of ICDE*. IEEE, 2011, pp. 780–791.

[43] A. Fourney, B. Lafreniere, P. Chilana, and M. Terry, "InterTwine: Creating interapplication information scent to support coordinated use of software," in *Proc. of UIST*. ACM, 2014, pp. 429–438.

[44] C. Liu, J. Wang, and Y. Han, "Mashroom+: An Interactive Data Mashup Approach with Uncertainty Handling," *Journal of Grid Computing*, vol. 12, no. 2, pp. 221–244, 2014.

[45] E. Santos, L. Lins, J. Ahrens, J. Freire, and C. Silva, "VisMashup: streamlining the creation of custom visualization applications," *Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 1539–1546, 2009.

[46] J. A. Cottam and A. Lumsdaine, "Automatic Application of the Data-State Model in Data-Flow Contexts," in *Proc. of IV*. IEEE, 2010, pp. 5–10.

[47] G. Di Lorenzo, H. Hacid, H.-y. Paik, and B. Benatallah, "Data integration in mashups," *ACM SIGMOD Record*, vol. 38, no. 1, p. 59, 2009.

[48] J. I. Fernández Villamor, J. Blasco Garcia, C. A. Iglesias Fernandez, and M. Garijo Ayestaran, "A semantic Scraping Model for Web Resources - Applying Linked Data to Web Page Screen Scraping," in *Proc. of ICAART*. SciTePress, 2011, pp. 451–456.

[49] B. Hartmann, S. Doorley, and S. R. Klemmer, "Hacking, mashing, gluing: Understanding opportunistic design," *Pervasive Computing*, vol. 7, no. 3, pp. 46–54, 2008.

[50] A. Moumtzidou, V. Epitropou, S. Vrochidis, S. Voth, A. Bassoukos, K. Karatzas, J. Moßgraber, I. Kompatsiaris, A. Karppinen, and J. Kukkonen, "Environmental Data Extraction from Multimedia Resources," in *Proc. of MAED*. ACM, 2012, pp. 13–18.

[51] M. Goebel and M. Ceresna, "Wrapper Induction," in *Encyclopedia of Database Systems*, L. Liu and M. T. Özsu, Eds. Springer, 2009, pp. 3560–3565.

[52] J. C. Bare, P. T. Shannon, A. K. Schmid, and N. S. Baliga, "The Firegoose: two-way integration of diverse data from different bioinformatics web resources with desktop applications," *BMC Bioinformatics*, vol. 8, no. 1, pp. 456:1–456:12, 2007.

[53] D. Tenenbaum, J. C. Bare, and N. S. Baliga, "GTC: A web server for integrating systems biology data with web tools and desktop applications," *Source Code for Biology and Medicine*, vol. 5, no. 1, pp. 7:1–7:3, 2010.

[54] M. S. Doderer, C. Burkhardt, and K. A. Robbins, "SIDECACHE: Information access, management and dissemination framework for web services," *BMC research notes*, vol. 4, no. 1, pp. 182:1–182:7, 2011.

[55] M. S. Doderer, K. Yoon, and K. A. Robbins, "SIDEKICK: Genomic data driven analysis and decision-making framework," *BMC Bioinformatics*, vol. 11, no. 1, pp. 611:1–611:12, 2010.

[56] M. Hogräfer and H.-J. Schulz, "ReVize: A Library for Visualization Toolchaining with Vega-Lite," in *Proc. of STAG*. Eurographics Association, 2019, pp. 129–139.

[57] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer, "Vega-Lite: A Grammar of Interactive Graphics," *Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 341–350, 2017.

[58] L. Nonnemann, *AnyProc*, 2020, accessed March 10, 2020. [Online]. Available: https://vcric.igd-r.fraunhofer.de/univa

[59] M. Aehnelt, H.-J. Schulz, and B. Urban, "Towards a Contextualized Visual Analysis of Heterogeneous Manufacturing Data," in *Advances in Visual Computing*, ser. Lecture Notes in Computer Science. Springer, 2013, vol. 8034, pp. 76–85.

[60] L. Nonnemann, M. Haescher, M. Aehnelt, G. Bieber, H. Diener, and B. Urban, "Health@Hand A Visual Interface for eHealth Monitoring," in *Proc. of ISCC*. IEEE, 2019, pp. 1093–1096.