Visual Methods for Analyzing Time-Oriented Data

Wolfgang Aigner, Silvia Miksch, Wolfgang Müller, Heidrun Schumann, and Christian Tominski

***Abstract*—Providing appropriate methods to facilitate the analysis of time-oriented data is a key issue in many application domains. In this paper, we focus on the unique role of the parameter time in the context of visually driven data analysis. We will discuss three major aspects – visualization, analysis, and the user. It will be illustrated that it is necessary to consider the characteristics of time when generating visual representations. For that purpose we take a look at different types of time and present visual examples. Integrating visual and analytical methods has become an increasingly important issue. Therefore, we present our experiences in temporal data abstraction, principal component analysis, and clustering of larger volumes of time-oriented data. The third main aspect we discuss is supporting user-centered visual analysis. We describe event-based visualization as a promising means to adapt the visualization pipeline to needs and tasks of users.**

***Index Terms*—Time-Oriented Data, Visualization, Analysis, User.**

I. Introduction & Motivation

CONSIDERING the characteristics of data is vital when designing visual representations. A salient characteristic is whether or not data are related to time. That time is an outstanding dimension is reflected by Shneiderman's Task by Data Type Taxonomy [1], where temporal data are identified as one of seven basic data types. Nowadays, time-oriented data are ubiquitous in many application domains as for example in business, medicine, history, planning, or project management. For a long time visual methods have been successfully applied to analyze such data. A wide repertoire of interactive techniques for visualizing datasets with temporal dependencies is available. However, many current visualization frameworks have not yet considered time as a special dimension, but rather as a common quantitative parameter. According to Thomas and Cook [2] it is in general a problem that *"Most visualization software is developed with incomplete information about the data and tasks. (. . .) New methods are needed for constructing visually based systems that simplify the development process and result in better targeted applications."*

In this paper, we point out challenges that arise when visualizing time-oriented data, and take a look at possible solutions to these challenges. To find solutions, it is absolutely mandatory to take into account the following three major aspects:

- Visualization,
- Analysis, and the
- User.

Wolfgang Aigner and Silvia Miksch are with the Danube University Krems, E-mail: {wolfgang.aigner, silvia.miksch}@donau-uni.ac.at; Wolfgang Müller is with the University of Education Weingarten, E-mail: muellerw@acm.org; Heidrun Schumann and Christian Tominski are with University of Rostock, E-mail: {schumann,ct}@informatik.uni-rostock.de.

Manuscript received Month XX, 200X; revised Month XX, 200X.

In Section II, we focus on visualization methods for time-oriented data. We will show that the term *time-oriented data* comprises several types of data with different meanings and applications. Designing or applying visual representations can only be successful if one is aware of these different types. This will be demonstrated with several examples of visualization techniques that stem from our own work or are available in literature.

Usually, time-oriented data are large – not only in terms of the number of data items, but also in terms of the number of observed attributes. Ordinary visualizations of such data can lead to overcrowded and cluttered displays, and are therefore of limited use. Data abstractions can help to gain insight even into larger datasets. This is the point where analytical methods come into play. In Section III, we will illustrate (again by examples) the usefulness of combining visual and analytical methods particularly related to time-oriented data.

In order to achieve better targeted applications, users and their tasks and needs must not be neglected, as it is still often the case in today's visualization tools. Apparently, interaction is a key to adapting visual and analytical methods to the user's task at hand. However, not all parameters are intuitive and easy to set. Particularly in cases where complex visual analysis processes have to be steered, having some form of user support or guidance turns out to be helpful. Section IV discusses how such a support can be realized. The basic idea is to find events in the data and to trigger automatic parameter adjustments.

In the last section (Section V), we will briefly recapitulate our discussions and derive possible directions for future work on visual analysis of time-oriented data.

II. Visualizing Time-Oriented Data

When we speak of time-oriented data, we basically mean data that are somehow connected to time. Certainly, this vague description is not sufficient when users have to choose or developers have to devise appropriate visualization methods. An essential requirement for achieving expressive and effective visualization is to consider the characteristics of the data to be presented, which in our case are particularly related to the dimension time. A lot of work has been done to formulate the notion of time in many areas of computer science, including artificial intelligence, data mining, simulation, modeling, databases, and more. A theoretical overview along with many references to fundamental publications is provided by Hajnicz [3]. However, as she points out, the terminology is not consistent across the different fields [3], and hence, does not integrate well with visualization. Therefore, we adapted the work of Frank [4], where he presents principal orthogonal design dimensions to characterize different types of times. The most important criteria from a visualization point of view are the following:

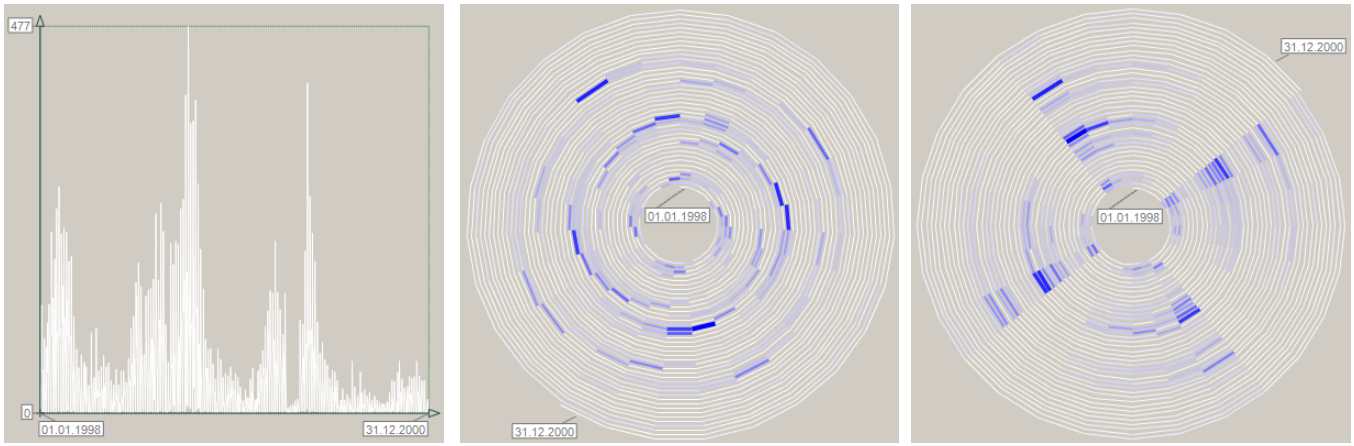

Fig. 1. Different visual representations of a time-oriented dataset describing the number of influenza cases over a period of three years – left: Time series plot (periodic pattern is difficult to discern), center: SpiralGraph encoding 27 days per cycle (improperly parameterized – periodic pattern is hard to see), right: SpiralGraph encoding 28 days per cycle (properly parameterized – periodic pattern stands out).

- *Linear time vs. cyclic time:* Linear time assumes a starting point and defines a linear time domain with data elements from past to future. On the other hand, many natural processes are cyclic, e.g., the cycle of the seasons. To represent such phenomena, a cyclic time domain can be applied. The ordering of points in a strictly cyclic time domain is meaningless with respect to a cycle, e.g., winter comes before summer, but winter also succeeds summer.
- *Time points vs. time intervals:* Discrete time points describe time as abstractions comparable to discrete Euclidean points in space. Time points have no duration. In contrast to that, interval time uses an interval scaled time domain like days, months, or years. In this case, data elements are defined for a duration, delimited by two time points. Both time points and time intervals are called temporal primitives.
- *Ordered time vs. branching time vs. time with multiple perspectives:* Ordered time domains consider things that happen one after the other. For branching time, multiple strands of time branch out, which facilitates description and comparison of alternative scenarios (e.g., for project planning). This type of time supports decision making processes where only one alternative will actually happen. Time with multiple perspectives allows more than one point of view at observed facts (e.g., eye-witness reports).

Since it is difficult to consider all of the mentioned aspects in a single visualization technique, the majority of available visualization methods address specific cases only – mostly the visualization of linear time dependencies. The approaches known in literature can basically be differentiated into techniques that visualize time-oriented data and techniques that visualize time per se. In the first case, the focus is set on representing data. Mostly quantitative, but also qualitative time-oriented attributes are represented with respect to a rather simple time axis (e.g., multivariate data represented with respect to linear time). The second case focuses on representing characteristics of the time domain and its temporal primitives, while only rather simple data representations are considered (e.g., Gantt charts to represent relations between time intervals).

It must be stressed that techniques developed for a particular time characteristic should not be applied to visualize data that exhibit different characteristics. Doing so can result in inexpressive or ineffective visual representations, and can lead to misunderstandings and false interpretations. To support the data analysis process via adequate visualization methods, it is therefore crucial to analyze the time characteristics of the dataset under investigation.

In what follows, we will illustrate the importance of choosing and parameterizing a visualization method properly with respect to given time characteristics. We will also give examples of visualization techniques that are suitable for different instances of Frank's taxonomy of types of times as presented before. Note that the considered time characteristics are used for illustrative purposes and cannot cover all aspects of the complexity of the dimension time. Frank's taxonomy encompasses more features and besides that, other taxonomies for characterizing time and visualization techniques for time-oriented data exist [5], [6], [7], [8]. We do not intend to provide a comprehensive overview on all aspects of the dimension time, but instead focus on the importance of considering the characteristics of time for an integrated visually driven data analysis.

a) Linear time vs. cyclic time: First, we point out the crucial influence of linear vs. cyclic time characteristic on the expressiveness of a visualization. Fig. 1 shows three different visual representations of the same time-oriented dataset, which contains the daily number of cases of influenza that occurred in the northern part of Germany during a period of three years. In the leftmost figure, a simple time series plot is used. Although peaks in time can be easily recognized when examining this representation, cyclic behavior of the data can only be guessed and it is hard to discern whether repetitive quantitative patterns in fact do exist. The representation is not particularly helpful in analyzing data with respect to cyclic temporal patterns.

The *Spiral Graph* [9] (see also [10], [11]) is a visualization technique that focuses on cyclic characteristics of time-oriented data by using a spirally shaped time axis (see Fig. 1

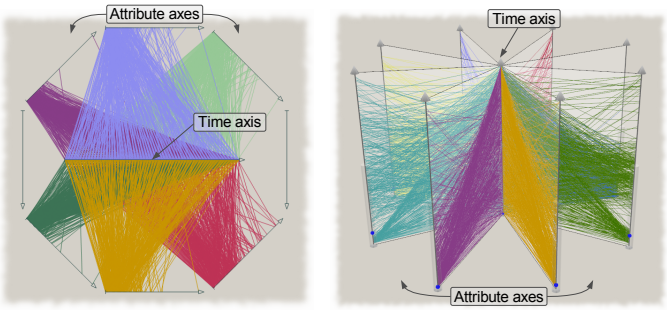

Fig. 2. TimeWheel – Multivariate time-oriented data represented using a TimeWheel – left: a 2D TimeWheel [12], right: the 3D analog [13].

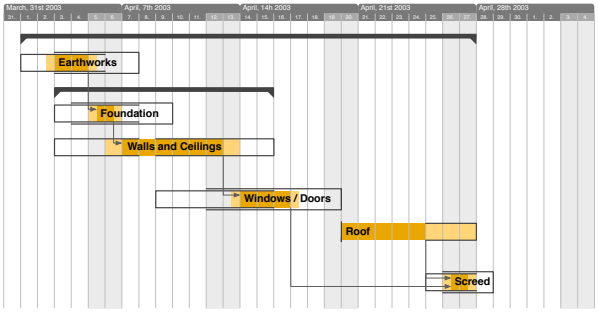

Fig. 3. PlanningLines [14] – Project plan represented using PlanningLines, which allow depiction of temporal uncertainties via special glyphs.

center and right). The main purpose of this technique is the detection of previously unknown periodic behavior of the data. This requires appropriate parameterization of the visualization method. The representation in the center of Fig. 1 is suited for cyclic time-oriented data, but it is improperly parameterized with a cycle length of 27 days; a pattern is not clearly visible. In contrast to that, the rightmost representation in Fig. 1 is adequately parameterized with a cycle length of 28 days, and immediately reveals a periodic pattern present in the analyzed data. The continuous differences of the number of cases between Sundays and Mondays are quite obvious. Apparently, that pattern will also be visible if the cycle length is set to 7 or 14 days.

Usually, it is difficult to find suitable parameter settings for unknown datasets. Therefore, it makes sense to support the detection of patterns either by applying analytical methods (see Section III) or by animating smoothly through possible parameter settings (i.e., different cycle lengths). In the latter case, periodic behavior of the data becomes immediately apparent by the emergence of a pattern. When such a pattern is spotted, the user stops the animation and an interesting cycle length has been found.

This discussion shows that not only selecting an appropriate technique is decisive for successful visualization, but also the proper parameterization of the chosen technique. This also implies that interaction facilities are needed to allow users to re-parameterize visualization methods according to their task at hand. Only then, visualization can take full advantage of the capabilities of the human perceptual system, e.g., in recognizing patterns and motion.

b) Time points vs. time intervals: Whether temporal attributes are conceptually modeled as time points or time intervals, is another important characteristic that influences the appropriateness of visualization methods.

Most of the known visualization techniques that represent time-oriented data consider time points. An example for a technique particularly suited for point-based time is the *TimeWheel* technique [12]. The TimeWheel is a multi-axes representation for visualizing multivariate data over time (see Fig. 2). This is achieved by putting a time axis to a prominent position in the center of the display. A set of axes that encode time-dependent attributes is circularly arranged around the central time axis. For each time point in the considered data, lines descend from the time axis to the corresponding points on each of the attribute axes. The TimeWheel can be rotated to bring different attributes into the focus. Furthermore, each axis can be equipped with a slider to zoom into value ranges of interest, and in particular, to navigate the time axis. Interactive labels can be activated on demand to facilitate the identification of data values. Since the TimeWheel uses lines to represent data for each point in time, it is useful only for multivariate data that are related to time points; data based on time intervals cannot be represented.

So far, we have mentioned techniques that visualize quantitative data values related to time points. Other approaches focus on representing temporal primitives and relations among them (e.g., *LifeLines* [15] to visualize personal histories, or the new metaphors for visualizing temporal queries introduced by Chittaro et al. [16]). A technique particularly suited to visualize temporal intervals (here used to model activities) and their uncertainties at a high level of detail are the *PlanningLines* [14]. PlanningLines consist of two encapsulated bars that represent minimum and maximum duration and are bounded by two caps representing start and end intervals (see Fig. 3). Apart from allowing the representation of possible distributions of start, end, and duration of an activity, a second important issue is addressed by PlanningLines – temporal uncertainty. Uncertainty might be introduced by explicit specification usually connected with future planning (e.g., "The meeting will start at 11 a.m. and will take approximately one hour" – which means that it is not quite clear when the meeting will be over) or is implicitly present in cases where data are given with respect to different temporal granularities (e.g., days vs. hours). PlanningLines support interactive zooming and panning, which is particularly useful for fine-grain plans with large time scales.

c) Ordered time vs. branching time vs. time with multiple perspectives: Although Frank's taxonomy [4] lists branching time and time with multiple perspectives as relevant types of time, most techniques for visualizing time-oriented data consider ordered time only.

An example of a visualization technique that assumes an ordered collection of time points is the *ThemeRiver* [17]. It represents the number of occurrences of particular news topics in print media. Each topic is displayed as a colored current that changes its width continuously as it flows through time. The overall image is a river that comprises all considered topics (see Fig. 4). The ThemeRiver provides an overview

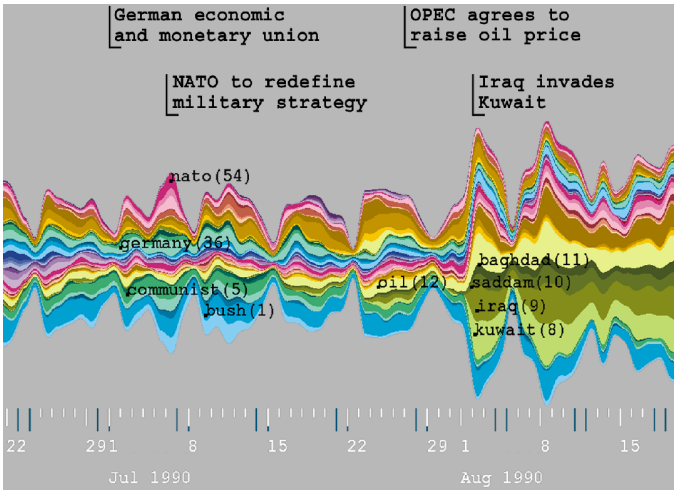

Fig. 4. The ThemeRiver [17] – The visual representation uses the metaphor of a river that flows through time. Currents within the river represent thematic changes in a document collection.

on what topics were important at certain points in time. Even though the ThemeRiver was originally invented to visualize thematic changes in document collections, it is also suitable to represent other quantitative data. In such cases, it is important to provide interaction techniques to rearrange the horizontal position of variables within the river. This is necessary because variables in the center of the river are perceptually emphasized, whereas variables represented at the rims of the river diminish in perceptibility.

The ThemeRiver as well as most visualization techniques known in literature are not suited to represent branching time or time with multiple perspectives. The few techniques for representing these types of time are capable of depicting only univariate qualitative data (e.g., Decision Chart [18] or PlanningLines [14]), or even visualize temporal primitives only; they cannot represent multiple time-oriented variables. Here, we see the need for advanced techniques to effectively visualize multivariate data exhibiting these specific time characteristics. This is an interesting direction for future work.

The bottom line of our discussion is that the characteristics of the parameter time have to be considered when creating visual representations of time-oriented data. We also indicated that integrating appropriate interaction methods is a key concern. Interaction is mandatory to allow users to re-parameterize a visual representation, and interaction is a must to facilitate different user tasks including navigation in time, directed and undirected search, comparison, and manipulation. Similar to what we said about visualization methods, interaction facilities also need to be user- and task-specific. For example, if the main task of a user is to compare multiple time-dependent variables, it makes sense to provide interaction techniques that allow navigating the time axis or brushing certain data values in different views (e.g., [19], [20]). In conclusion, only an adequately chosen and properly parameterized visualization technique in combination with user- and task-specific interaction methods can fully support the development of insight into time-oriented data.

III. Analyzing Time-Oriented Data

In the preceding section, we have indicated that choosing appropriate techniques, parameterizing them correctly, and incorporating useful interaction methods are essential requirements to achieve expressive and effective visual representations. When dealing with large volumes of data, additional analytical methods have to be included to derive higher levels of abstraction of the data. A large variety of time-series mining techniques have been developed in recent years[1]. Applying these techniques facilitates the interactive exploration of even huge datasets by starting with a compact overview image, which avoids overlapping of data, and then adding more details interactively [21].

From a visualizer's perspective, this fundamental procedure is expressed in Keim's *Visual Analytics Mantra* [22]: *"Analyze First - Show the Important - Zoom and Filter, and Analyze Further - Details on Demand."* Indeed, developing methods that fully adhere to this mantra (i.e., tightly integrate time-series mining and visualization) is a challenging task for future research.

In what follows, we describe our experiences in integrating visual and analytical methods. We will illustrate the usefulness of Keim's mantra by three examples: the concept of *temporal data abstraction*, *principal component analysis*, and *clustering*. These concepts address different concerns. Temporal data abstraction reduces value ranges from quantitative values to qualitative values, which are much easier to understand. PCA reduces the number of variables by switching the focus to major trends in the data. Clustering methods reduce the number of data tuples by finding expressive representatives for groups of tuples.

A. Temporal Data Abstraction

Temporal attributes are an important aspect in high-frequency domains or domains where heterogeneous data are present (e.g., the medical domain, observing human activities and behavior, or environmental monitoring). The big question is how huge volumes of continuously assessed data can be analyzed to ease further decision making. On the one hand, the data are too large to be interpreted all at once. On the other hand, the data are more erroneous than usually expected and some data are missing too. One possibility to tackle these problems is to apply knowledge-based techniques to derive *qualitative values or patterns* of current and past situations, called *data abstraction* – a term originally introduced by Clancey in his classical proposal on heuristic classification [23]. The objective of data abstraction in general is *"to create an abstraction that conveys key ideas while suppressing irrelevant details"* [24]. The basic idea is to use qualitative values or patterns, rather than raw data, for further analysis or visualization processes [25]. This helps in coping with the complexity of these processes. To compute data abstractions, several tasks must be conducted (e.g., selecting relevant information, filtering out unneeded information, performing

[1]A review of the vast body of work in time-series mining is beyond the scope of this paper. A valuable source for more information is http://www.cs.ucr.edu/~eamonn/TSDMA/ (accessed March 2007).

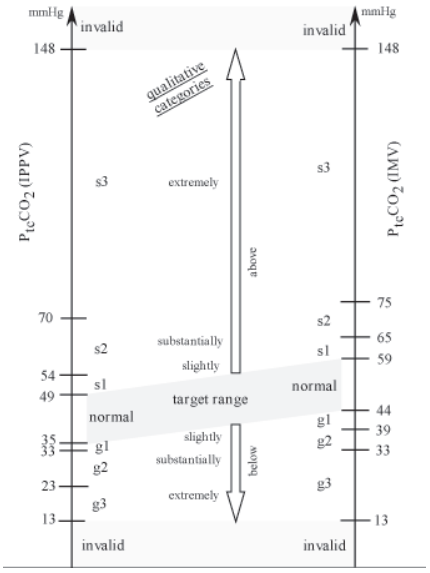

Fig. 5. VIE-VENT's schemata for data-point transformation [26] of $P_{tc}CO_2$ during intermittent positive pressure ventilation (IPPV, left) and intermittent mandatory ventilation (IMV, right). The qualitative data point categories are given in the middle column. For example, a $P_{tc}CO_2$ value of 60 will be transformed to "substantially above target range (s2)" during IPPV and to "slightly above target range (s1)" during IMV.

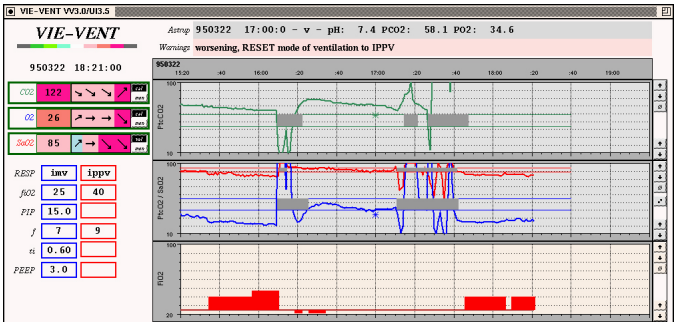

Fig. 6. VIE-VENT [26] – The user interface of VIE-VENT. The left-hand side region shows the blood gas measurements, their corresponding qualitative temporal abstraction on the top and the actual and recommended ventilator settings below. The right-hand side region gives plots of the most important variables over the last four hours (e.g., transcutaneously assessed blood gas measurements and ventilator settings).

calculations, sorting, and clustering). The consequent next step is to provide techniques to visualize data abstractions in a user- and task-specific manner.

Temporal data abstraction represents an important subgroup where the processed data are time-oriented. We distinguish *basic temporal abstraction* methods (e.g., state, gradient, and rate) and more *complex temporal abstraction* methods. The basic abstraction *state* corresponds to a classification (or computational transformation) of data values. *Gradient* corresponds to the sign of the derivative of a data value, and *rate* complies with the magnitude of the derivative during an interval (e.g., abstractions: *high*, *decreasing*, and *fast* for a temperature variable). Basic temporal data abstractions alone are not always sufficient to deal with time-oriented data, because these abstractions are unable to tackle shifting contexts, different expectations concerning the development of variables, or detection of more complex patterns. Higher-order temporal abstraction methods are needed to derive unified qualitative values and patterns. Therefore, we have investigated methods of complex temporal abstraction.

VIE-VENT [26] addresses context-sensitive and expectation-guided temporal abstraction methods in a medical application domain. The developed methods incorporate knowledge about data points, data intervals, and expected qualitative trend patterns to arrive at unified qualitative descriptions. They are based on context-aware schemata for data point transformation (see Fig. 5) and curve fitting to express the dynamics of and the reaction to different degrees data abnormalities. Smoothing and adjustment mechanisms are used to keep qualitative descriptions stable in case of shifting contexts or data oscillating near thresholds. For example, during intermittent positive pressure ventilation (IPPV), the transformation of the quantitative value $P_{tc}CO_2 = 56mmHg$ results in a qualitative $P_{tc}CO_2$ value of "substantially above target range"[2]. During intermittent mandatory ventilation (IMV) however, $56mmHg$ represent the "target value". Qualitative $P_{tc}CO_2$ values and schemata of curve fitting are subsequently used to decide if the value progression happens too fast, at normal rate, or too slow (see Fig. 6).

Qualitative descriptions and patterns as derived by temporal abstraction methods are heavily data dependent. The methods developed in the VIE-VENT system are one way to deal with cases of oscillating data where abstractions and hence interpretations are frequently changing. Another solution is presented in the *The Spread* [27]. It implements a time-oriented data abstraction method to derive steady qualitative descriptions from oscillating high-frequency data. We distinguish the following steps of processing and abstracting the data:

1) *Eliminating data errors*. Sometimes up to 40% of the input data are obviously erroneous, i.e., exceed the limits of plausible values.
2) *Clarifying the curve.* Transform the still noisy data into a steady curve with some additional information about the distribution of the data along that curve.
3) *Qualifying the curve.* Abstract quantitative values to qualitative values like "normal" or "high" and join data

[2] $P_{tc}CO_2$ = transcutaneous partial pressure of carbon dioxide

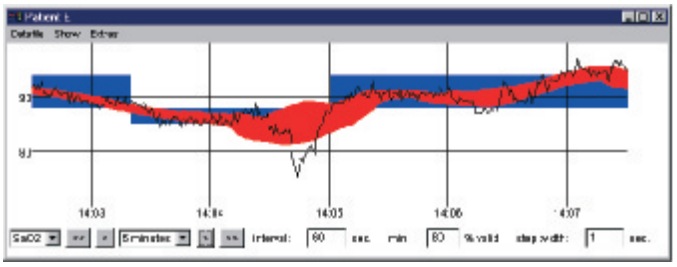

Fig. 7. The Spread [27] – The thin line shows the raw data. The red area depicts the *Spread*, the blue rectangles represent the derived temporal intervals of steady qualitative values. Increased oscillation leads to increased width of the spread, but not to a change of the qualitative value. The lower part of the figure shows the used parameter settings.

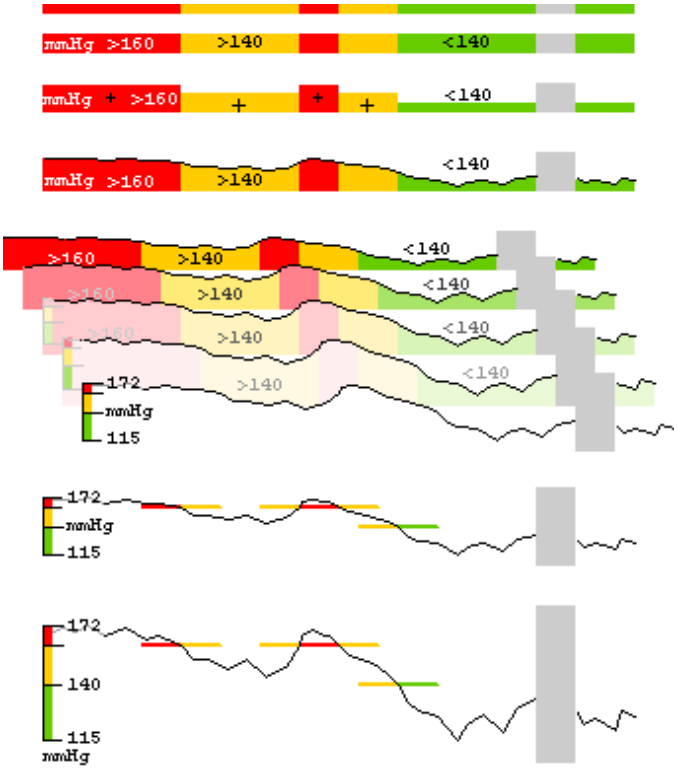

Fig. 8. Midgaard [28] – Steps of resizing/zooming the representation of a data stream from a broad overview with qualitative values to the fine structure with quantitative details (top to bottom).

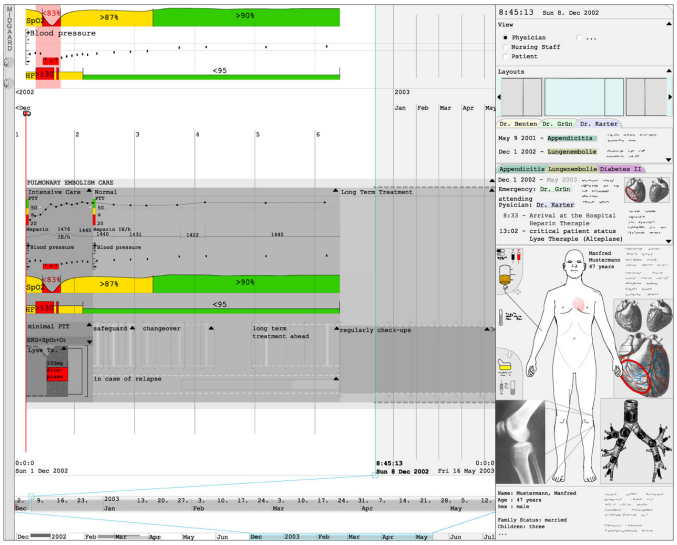

Fig. 9. The user interface of Midgaard [28] – The upper left part shows different measurements (e.g., blood gas measurements, blood pressure) and their corresponding temporal abstractions. The right part explains additional patient's information and the lower left part explains the time axis interaction: the selected subrange at the bottom time axis can be moved and rescaled to pan+zoom the time range shown in the middle and top time axes.

points with equal qualitative values to time intervals.

The Spread provides parameters to adjust the abstraction process (e.g., length of time window, permitted gaps, or points of changing the qualitative value). As an example, consider a physician who is observing continuously assessed measurements and wants to find time intervals of different qualitative regions like *"$P_{tc}CO_2$ is high for 5 minutes"*. When looking at the raw data, which typically oscillate, the physician will certainly have difficulties in finding reasonably long time spans with stable values. The Spread is able to support the physician in making qualitative assessments of the time intervals she is interested in (see Fig. 7).

Temporal abstraction methods as provided in VIE-VENT and The Spread are generic methods that can be used for different purposes. In the *Midgaard* project [28] these methods have been extended by several visualization techniques to enhance the understanding of qualitative and quantitative characteristics of a given time-oriented dataset. The challenges were not only to support the user in exploring the data with different tasks in mind, but also to capture as much temporal information as possible on a limited display space without loss of overview and details. We provide different levels of abstractions for time-oriented data. Switching between these levels results in a smoothly integrated semantic zoom functionality (see Fig. 8 and left-hand side of Fig. 9). Our methods were designed to allow users to interact with data and time (e.g., browsing and searching the time axis). The visualization of temporal aspects comprises three linked time axes (see Fig. 9). The first one (bottom) provides a fixed overview of the underlying data and their full temporal range. Selecting a subrange in that time axis defines the temporal bounds for the second (middle) and the third (top) time axis. By interactively adjusting the subrange users can easily *zoom and pan* in time.

The described basic and complex temporal abstraction methods are very useful in tackling the complexity of analyzing and interpreting huge volumes of time-oriented data. We have explored the usefulness of our methods by cooperating with medical experts, who found it easy to capture severe or stable health conditions of patients. Moreover, these abstractions can be used for further reasoning or in guideline-based care for a simplified representation of treatment plans.

Using data abstraction is more than ever a current research topic [29]. The advantage of abstract descriptions or patterns is their unified applicability in various applications scenarios, regardless of the origin of the data to be visualized.

B. Principal Component-Based Analysis

As already mentioned, time-oriented data are often of multivariate nature. *Principal component analysis* (PCA) [30] is a technique frequently applied to reduce the number of variables and to detect structure in multivariate datasets [31]. As such, PCA represents another approach to data abstraction. Different to the previously discussed approaches, which work on the original data space to derive qualitative data abstractions, PCA results in a transformation of the original data space into a different domain – the *principal component space*. The goal of this transformation is to make important trends in the data directly accessible.

The extraction of *principal components* (PCs) amounts to a variance-maximizing rotation of the original variable space. That is, the original data space is transformed in such a

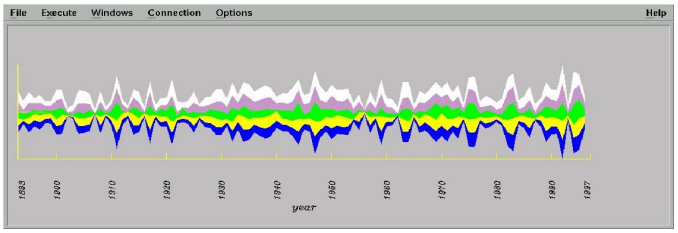

Fig. 10. Visualization of a climate dataset using a *ThemeRiver* [32] approach. The graph depicts five time-dependent variables: summer warmth (blue), summer days (violet), hot days (green), summer mean temperature (yellow), and mean of extreme (white) for a period of more than 100 years.

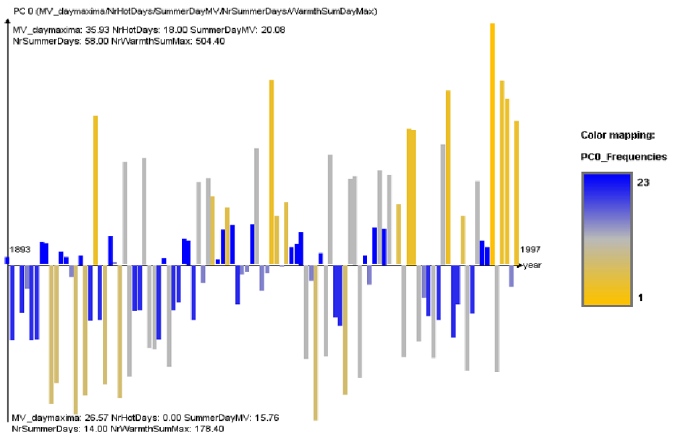

Fig. 11. Bar chart visualization of PC0 over time for the dataset from Fig. 10. Upward bars represent warmer conditions, whereas downward bars stand for colder summers (but not necessarily negative temperatures). Frequencies of data values are mapped onto color to further distinguish typical (blue) and outlier (orange) years. Major trends are clearly visible: The first third of the time line is dominated by average warm summers mixed with the coldest summers; hot summers occur followed by cold summers in the end of this period; in general, outlier summers cumulate at the end of the time line.

way that the first PC resembles most of the original dataset's variance, the second PC most of the remaining variance, and so on. Identifying these factors leads to a more compressed description of correlations in the data, and thus, to a better understanding of underlying features and trends. Since the PCA provides PCs ordered by their significance, it also offers an excellent basis for dimension reduction in case of multidimensional data. Less relevant factors can be omitted leading to a lossy, but more compact representation.

In principle, PCA does not distinguish between independent and dependent variables in this process: all variables are weighted and handled equally. As mentioned before, this often raises problems in the context of time-oriented data. In particular, the temporal context gets lost and the interpretation gets hampered. Therefore, it is preferable to exclude the independent variable "time" from PCA. Time and computed PCs should be rejoined to restore the temporal context afterwards.

To demonstrate the strengths of combining PCA with visualization we will take a look at a simple example. The data we consider is related to climate research. The basis of the example is a meteorological dataset that contains daily observations of temperature, precipitation, wind, air pressure, and others for a period of more than 36,500 days (100 years). To analyze the development of global warming over the last century, we cooperated with climate researchers to derive a dataset that focuses on summer weather conditions only [33]. That condensed dataset is on a yearly basis and comprises five variables: summer warmth (sum of max temperatures for days with $T_{max} \geq 20^o$C), summer days (number of days with $T_{max} \geq 25^o$C), hot days (number of days with $T_{max} \geq 30^o$C), summer mean temperature (mean of daily average temperature T_{avg}), and mean of extreme (mean of daily max temperatures T_{max}). All five are quantitative variables that either count days with specific weather conditions or contain aggregated temperature information; their strong correlation has been intended by the climate researchers involved.

The condensed dataset can be visualized with a *ThemeRiver* (see Fig. 10). In this graph, constrictions in the river stand for low data values, which indicate particularly cold summers. Broad flow snapshots characterize particularly hot summers. On first impression, a general overview and important characteristics of the dataset are depicted well.

We will now show how PCA and an additional simple bar chart representation can help to derive further information from the data. To find major trends in terms of climate change, PCA was applied to the condensed dataset. Time was excluded from the analysis to retain the temporal context. The bar graph in Fig. 11 depicts the first PC only (i.e., PC0), to which all variables contribute. Bars above the time axis represent hot summers, whereas bars below the time axis stand for colder summers. Additionally, a color-coding of PC frequencies was added to enhance expressiveness: Orange bars represent outliers, whereas blue bars represent more common conditions (the colors are not related to temperature). The combination of PCA and simple visualization succeeds in presenting major trends in the data very clearly: Average warm summers dominate the first third of the century, containing also the coldest summers (orange bars below the time axis). Hot outlier summers cumulating at the end of the century can also be detected very easily (orange bars above the time axis). Moreover, two additional converse trends can be identified: Hot summers occur followed by colder summers in the end of this period. In the last third hot summers preponderate, with the warmest summers at all. The PC visualization in Fig. 11 depicts corresponding trends very well. This demonstrates the value of PC-based temporal abstractions in the visual analysis of time-dependent data. Nonetheless, one should recall that our condensed climate dataset represents a special case where all variables are strongly correlated. That correlation is the reason why PC0 separates warm and cold summers so well. When analyzing arbitrary temporal datasets, further PCs may be necessary to describe all trends. In such cases, not only more responsibility of the user is required, but also flexible mechanisms and controls are needed to determine variables that should be considered for PCA and to select PCs that should be visualized. This calls for an integration of analytical analysis and visualization in a single tool.

As mentioned above, PCA represents an almost completely automatic approach for temporal data abstraction. The advantage is that a user can get an abstracted view on the data very easily. Nonetheless, it is sometimes hard to relate

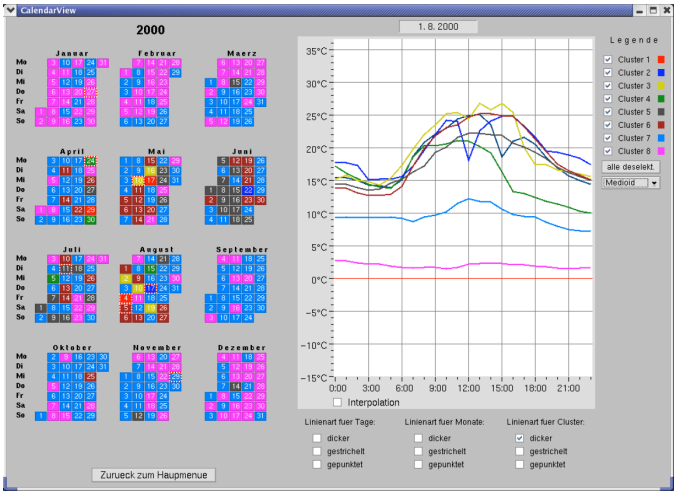

Fig. 12. *Cluster Calendar View* [36], [37] – The plots show cluster representatives as daily temperature profiles. The calendar view illustrates by color which days belong to which cluster, i.e., show similar profiles.

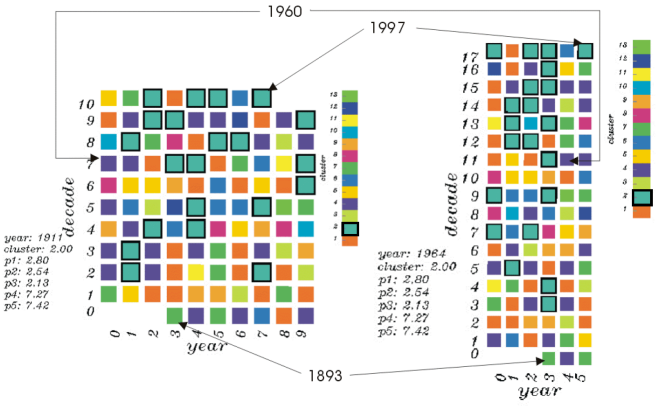

Fig. 13. *Rectangular View* – Visualization of a temporal clustering of meteorological time-oriented data from the Potsdam observation station. Thirteen clusters of yearly temperature curves have been extracted from the data. In this example, changing the periodicity (denoted as decade) from 10 years (left) to 6 years (right) helps in identifying a temporal pattern for cluster 2 (see [37]).

patterns visible in PC space to original data variables and the abstracted views are not always easy to interpret. What can help in these cases are approaches to enhance readability of PC-based diagrams by incorporating additional information or interactive means to support relating PCs to original data [34]. Still, there is room to improve the expressiveness of PC-based visualization in further research.

C. Clustering

After discussing temporal data abstraction and dimension reduction with PCA, we now want to take a closer look on data aggregation. Clustering methods provide a basis for this purpose. Clustering relates to partitioning a dataset into subsets exhibiting a certain similarity. The clustering process also provides an abstraction of the data. Concentrating on the *clusters*, rather than on individual data values allows for an analysis of datasets with a much larger number of tuples. Appropriate *distance or similarity measures* lay the ground for clustering. Distance and similarity measures are profoundly application dependent. This has lead to a large number of different measures and clustering algorithms [35]. Selecting appropriate algorithms is typically difficult. Careful adjustment of parameters and regular validation of the results are also essential tasks in the process of clustering. Different to PCA, the variable "time" is typically included in the clustering process to reveal clustering with respect to temporal aspects. The resulting clustering may also lead to a temporal data abstraction.

Visualization has been frequently applied to validate and guide the clustering process. Different mining tools provide cluster algorithms and techniques to visualize the clustering results. However, most of the techniques for visualizing clusters neglect the temporal context, thus making it difficult to analyze data with respect to fundamental time-oriented tasks (e.g., to associate data values and clusters with particular time steps).

A technique specifically designed for the analysis of clustered time-oriented data is the *Cluster Calendar View* [36] (see Fig. 12). It applies a calendar metaphor to represent the temporal context. Cluster affiliation is presented indirectly by color-coding. A line plot presents details on trends subsumed in selected clusters. Fig. 12 shows an example in the context of meteorological data. In this example, clusters 7 (light blue) and 8 (magenta) represent typical daily temperature curves and hence dominate the calendar. All the other clusters are more or less atypical and represent outliers. Furthermore, the color-coded calendar allows to reveal fast changes in cluster sequences for example in the first part of August. Brushing techniques provide additional support in the exploration process. For instance, we can highlight clusters that are similar to a selected cluster. The Cluster Calendar View facilitates comparison of cluster representatives (overview), exploration of the values of a single cluster representative (abstract detail), and exploration of daily and monthly values of interest (specific details).

In contrast to the Cluster Calendar View, the *Rectangular View* [37] depicts cluster information directly, thus allowing for the display of data for much larger time frames. The Rectangular View utilizes a tablet-like layout to present clusters as well as cluster centroids. Each cluster is visualized as a color-coded square. Clusters are positioned on the tablet from the lower left to the upper right with respect to their temporal location. Various interaction techniques extend the functionality. Temporal brushing allows to focus on specific time steps. Interactive modification of the cluster arrangement helps in detecting and understanding temporal patterns. In Fig. 13, for instance, a certain periodicity of cluster 2 can be observed when placing 6 years per row instead of 10. While this cluster appears frequently in columns 1 to 3, it is less existent in all other columns (0, 4, 5). The implication of a quasi-6-year cycle leads to new explanations and models on the transition from stable climatic states to new ones for the previously introduced meteorological dataset.

In this section, we demonstrated the usefulness of analytical methods to gain insight into larger volumes of time-oriented

data. Temporal data abstraction aims at gaining qualitative high-level insights. Principal component analysis and clustering help in handling larger numbers of variables respectively tuples in time-oriented data. All three methods applied to large time-oriented datasets provide different levels of abstraction and help to reveal major trends in the data.

Many more time-series analysis methods are known in literature. The information gained by these methods can be utilized to further support different steps in the analysis and visualization process to provide additional guidance to users. For instance, Seo et al. [38] and Müller et al. [34] present interactive techniques for data selection and attribute mapping based on information from clustering and PCA; Keogh et al. [39] integrate mining methods to drive interactive visual exploration of time-series.

IV. User-centered analysis via events

The methods presented in the previous sections are useful tools to facilitate visualization and analysis of time-oriented data. We already indicated that this is true only if the methods are parameterized according to the users' needs and tasks. This brings us to the third major point of our discussion – the user. User interaction is a way to manually parameterize the described visualization and analysis tools. Many tools provide an interactive graphical user interface to adjust the parameters of analytical methods (e.g., via sliders or check boxes). Visualization views can usually be adjusted via common view navigation (zoom, pan, rotation) [40], dynamic queries [39], and brushing [20].

However, it is not always easy for users to find parameter values that suit the analysis task at hand. Particularly analytical methods often have parameters that are not self-explanatory, and hence, are not easy to set. Moreover, the increasing complexity of visualization methods makes it more difficult for users to parameterize the visualization properly. What is needed is some form of support that helps in steering the visual analysis. A promising concept that addresses the automatic parameterization of visual representations is *event-based visualization* [41]. The thought behind this concept is to gain benefit from incorporating visualization and event methodology. Commonly, events are considered happenings of interest that trigger some automatic actions. This concept is prevalent in various application fields, including active databases, software engineering, and software visualization.

In our understanding, events occur if user-defined conditions, which are expressed with respect to entities of a dataset, become true. The basic idea of event-based visualization is to let users specify their interests as *event types* (i.e., encapsulations of conditions), to determine if and where these interests match in the data (i.e., detect *event instances*), and to consider detected event instances when generating the visual representation. This basic procedure requires three main steps - *1) event specification*, *2) event detection*, and *3) event representation*. We will give detailed descriptions on each of these steps in the next paragraphs. Fig. 14 illustrates how event-related components can be attached to the visualization pipeline (see [42]), which internally comprises data analysis, filtering, mapping, and rendering. Data analysis and filtering can be realized by the methods presented in Section III. How time-oriented data can be mapped (and rendered) to graphical representation was shown in Section II.

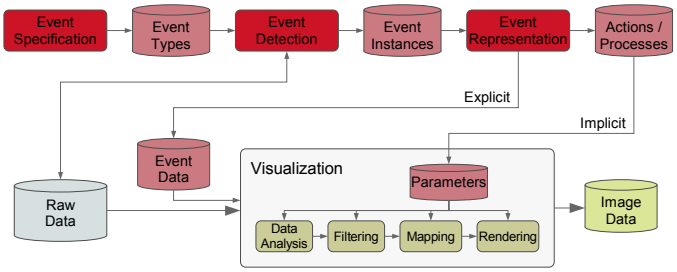

Fig. 14. Model of event-based visualization – The figure shows the major steps of event-based visualization (event specification, event detection, and event representation) attached to the well-known visualization pipeline.

1) Describing user interests: The *event specification* is the step where users describe their interests. To be able to find actual matches of user interests in the data, the event specification must be based on formal descriptions. For this purpose, event formulas have been developed. These formulas make use of elements of predicate logic, including variables, predicates, functions, aggregate functions, logical operators, and quantifiers. The elements may be used in a well defined way to create valid event formulas. We consider different variants of event types to facilitate the specification of interests with respect to relational datasets. Tuple event types can be used to detect interesting data tuples (e.g., tuples that show an exceeded threshold) and attribute event types are useful for finding attributes of interest (e.g., attribute with the highest average value). For an analysis of time-oriented data, this alone is not sufficient. Therefore, sequence event types are also supported. They enable users to specify conditions of interest regarding temporally ordered sequences of tuples (e.g., sequence of days with rising stocks). Sequence event types extend the existing event formulas with sequence-related notations (inspired by Sadri et al. [43]). A combination of event types to composite event types is also possible. They are realized via set operators. Because we rely on extended predicate logic and set theory, the expressiveness of the introduced event types is limited to these formalisms. However, the model of event-based visualization is not limited to certain fixed event types, but can be extended with event types as required for particular application contexts.

To give a simple example of a sequence event type, we will formulate the following interest: *"Find three successive days with increases of more than 15% in the number of influenza infections."* This interest is expressed as $\{(x,y,z)_{date} \mid z.flu \geq y.flu * 1.15 \wedge y.flu \geq x.flu * 1.15\}$. The first part of the formula defines three variables $(x,y,z)_{date}$ that are sequenced by date. To express the condition of interest, these three variables are set into relation using predicates, functions, and logical connectors.

Certainly, common users will have difficulties in describing their interests by using event formulas directly. To facilitate the specification of interests as formal event types, we developed

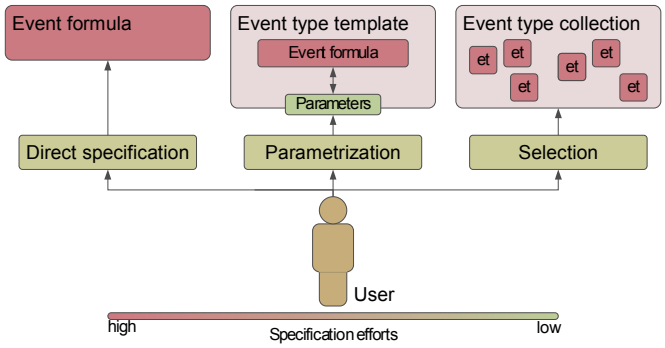

Fig. 15. User-centered event specification model – Event types can be specified by using event formulas directly (left), by parameterizing event type templates (middle), or by selecting from a predefined application-specific collection of event types (right). The effort required for event specification decreases in the same order.

a model for user-centered event specification. This model provides expert, common, and less-experienced visualization users with different specification methods. The three different levels of the model are *direct specification*, *specification by parameterization*, and *specification by selection* (see Fig. 15).

Although the model is based on the described event formulas, the complete functionality of these formulas is available only to expert users at the level of direct specification.

To ease the event specification for common users, so called event type templates are provided. Basically, the idea was to hide the complexity of event formulas from users. Event type templates use an internal event formula that cannot be changed directly, but can be adjusted to the users' needs via easy to set parameters. An example of an event type template is a threshold template where the two parameters "threshold" and "variable" can be set by users. An event instance is detected once the chosen variable exceeds the set threshold. Templates are particularly useful to encapsulate sequence event types. Interests like an increase of a "variable" over a certain "period of time" can be easily adjusted to the task at hand without typing entirely new event formulas.

The third level of event specification is based on simple selection. The event specification by selection addresses not only less-experienced visualization users, but also users (e.g., managers) who seek quick access to relevant information contained in the data to be analyzed. The idea is to provide a collection of expert-defined event types that are particularly tailored to the application context. In the case of time-oriented data, visualization tasks like identification of certain values in time or detection of behavioral patterns (e.g., the aforementioned increase in cases of influenza) could be formulated by domain experts. Predefined event types must be assigned with expressive labels and descriptions, so that users can easily select the event types they are interested in. It is also helpful to enhance the event collection with a semantic structure (e.g., by grouping the collection with respect to different user tasks). Again, to devise such a semantic structure and to describe it expressively is a task for domain experts.

2) Finding relevant data portions: The *event detection* step determines whether the interests defined as event types are present in the dataset under consideration. Conducting the event detection results in a set of event instances, which describe where in the data interesting information is located. That is, entities that comply with user interest are marked as event instances. For event detection, the variables used in event formulas are substituted with concrete entities of the dataset (tuples, attributes, or sequences of tuples). In a second step, predicates, functions, and logical connections are evaluated, so that the event formula as a whole can be evaluated to either true or false. Since this procedure is very costly in terms of computation time, efficient methods must be utilized for the event detection. For detecting interesting tuples and attributes, capabilities of relational database management systems can be utilized. The detection of sequence events makes use of the OPS algorithm [43], which has proved to be efficient for querying sequenced data. If dynamic data (i.e., data that change over time) have to be considered, detection efficiency becomes crucial. Here, incremental detection methods can help. Such methods operate on a differential dataset, rather than on the whole data. However, incremental methods also impose restrictions on possible event types.

3) Considering user interests in visual representations: The last important step of event-based visualization is the *event representation*. The goal of this step is to incorporate detected event instances (which reflect the interests of the user) into visual representations. We identified three requirements that have to be accomplished in this regard:

1) Communicate the fact that something interesting has been found.
2) Emphasize interesting data among the rest of the data.
3) Convey what makes the data interesting.

The most important requirement is that the visual representation must reflect that something interesting is contained in the data. This is essential for event-based visualization of time-oriented data. To meet this requirement, easy to perceive visual cues (e.g., a red frame around the visual representation, exclamation marks, or annotations) are used. Alpha blending can be applied to fade out past events. The second requirement aims at emphasizing those parts of the visual representation that are of interest. Additionally, the visualization should communicate what makes the highlighted parts interesting (i.e., what is the particular event type). However, facing arbitrarily definable event formulas, this last requirement is difficult to accomplish.

We distinguish two basic possibilities for representing events. On the one hand, it makes sense to visualize event instances, rather than the whole dataset. In this way, the focus is set exclusively on the interests of the user. Since the number of events is usually smaller than the number of data items, even large datasets can be analyzed (certainly, the same holds true for principal components and clusters as presented in Section III). This way of representing events is referred to as *explicit event representation*. On the other hand, adjusting the parameters of visual representations according to occurred event instances is a promising alternative. By pursuing what we call *implicit event representation*, we can automatically set visualization parameters according to interests detected in the data. If we assume that user interests are related to user tasks and vice versa, implicit event representation can

help to achieve better targeted visual representations. The big challenge is to meet the above stated requirements merely by adapting visualization parameters. Apparently, availability of adequate visualization parameters is a prerequisite for implicit event representation.

To illustrate the potential of event-based visualization, we will discuss an example. We assume a user who has to search time-dependent human health data for uncommonly high numbers of cases of influenza. The task at hand is to detect where in time these situations have occurred. A possible way to accomplish this task is to use the TimeWheel technique [12]. However, without event integration the user will be provided with a TimeWheel that uses a standard parameterization (see Fig. 16(a)). The standard view shows influenza on the upper left axis (light green), time is represented on the central axis. Alpha-blending has been applied by default to reduce visual clutter. From the TimeWheel in Fig. 16(a) one can only guess from the labels of the axis showing influenza that there are higher numbers of cases; the alpha-blending made the particular lines almost invisible (see question mark). Several interaction steps are necessary to re-parameterize the TimeWheel to accomplish the task at hand.

In contrast to that, in an event-based visualization environment, the user can specify the interest *"Find days with a high number of cases of influenza."* as an event type ($\{x \mid x.flu \geq 300\}$) to be considered for the current analysis task. The event type can be stored and may be reused in further visualization sessions or by other users. If a new dataset is opened or if new tuples are added dynamically to a time-oriented dataset, the event detection is run to determine whether or not the data conform to the condition expressed in the event type. If this is the case, event instances are created for those data portions that fulfill the condition. To reflect the interest of the data analyst, i.e., to provide an individually adjusted TimeWheel, the parameters of the visual representation have to be altered. Parameter changes can be implemented either as instantaneous *actions* or gradual *processes* (e.g., smooth animation). In our particular example, we use an action that switches color and transparency of line segments representing event instances. Days with high numbers of influenza cases are excluded from alpha-blending and are drawn in white color. Additionally, the TimeWheel is rotated (as a whole) such that the axis representing influenza is moved gradually to an exposed position. The application of a gradual process is important in this case to support users in maintaining their mental map of the visual representation. The result of applying parameter changes as response to event instances is depicted in Fig. 16(b). This figure illustrates that event-based visualization eases the visual analysis of time-oriented data significantly, since the visual representation is adapted to the current visualization task. In the example, the identification of days with higher numbers of influenza infections is easy.

As the previous example indicates, considering user interests helps to achieve better targeted visual representations. By combining event-based methodology with visualization approaches, we give users the opportunity to describe their interests. The described event types address not only tuples and attributes of relational data, but also sequences of tuples, which are important when dealing with time-oriented data. By using predicate logic, a high level of flexibility is achieved; a wide range of concrete event types can be imagined. It must also be mentioned that our approach has been developed to support directed search, i.e., users know what they are looking for. Being aware of what users are interested in, we are able to automatically generate visualizations that are potentially more helpful for the users' task at hand than standard representations. By focusing on relevant parts of the data, we also achieve another level of data abstraction.

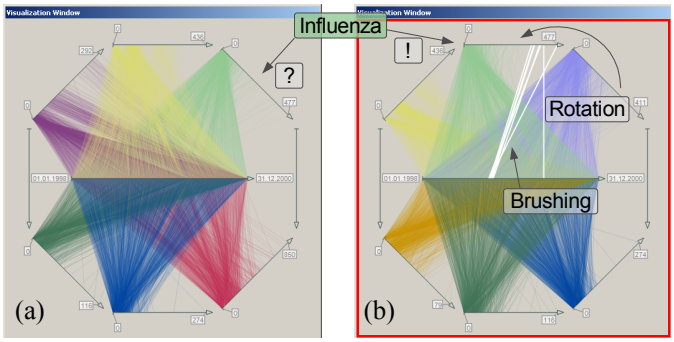

Fig. 16. Standard vs. automatic parameterization of a TimeWheel – (a) TimeWheel representing a time-dependent health dataset; the interests of the user are not considered in the standard parameterization, which aims at showing main trends. (b) TimeWheel representing the same data; the user's interests were recognized in the data and have been emphasized via highlighted lines and automatic rotation; the presentation is better targeted for the user's task at hand.

Until now, event-based visualization is not suited to automatically mine potential events in time-oriented data, i.e., to support undirected search, where users have no hypotheses about the data. With a tighter integration of visual and analytical methods, it should be possible to alleviate this concern. A second challenge for future work is to find general guidelines on how to realize parameter changes that indeed highlight event instances. Because the parameter space of visualization methods is usually very large and contains many interdependencies, we have to apply sophisticated methods (e.g., as suggested by House et al. [44]) to find and test appropriate parameter settings.

V. Conclusion

In this paper, we have investigated the role of time-oriented data in the context of visually driven data analysis. We have elaborated on the importance of choosing and parameterizing visualization techniques and interaction functionality properly with respect to characteristics of the time domain present in the data. However, in the light of huge datasets, visualizing all data in a comprehensible manner without burying possibly important information becomes more and more challenging. This challenge can be dealt with by conducting additional data analysis steps; many time-series analysis approaches are known in literature. By the examples of temporal data abstraction, PCA, and clustering, we have illustrated that analytical methods support the identification of the important in time-oriented datasets. The third question we addressed

concerns the integration of the user into the visual analysis process. We detailed on an approach to emphasize relevant information, called event-based visualization. This approach is mainly task-driven and aims at generating better targeted visual representations of time-oriented data (e.g., by automatic highlighting of relevant data as well as hiding of less relevant data).

Nonetheless, much more work has to be conducted in the future to support a comprehensive visual analysis. This includes the development of expressive visualization techniques for all kinds of time-oriented data. Especially multivariate data in the context of non-linear time domains as well as interval-based data and temporal uncertainties have to be considered to an increasing degree. A particularly challenging problem is to find new ways of describing tasks of the visual exploration process and to automatically adapt the whole analysis procedure according to the tasks at hand. This also includes specific interaction functions for investigating time-dependencies. For example, Doleisch et al. introduce different brushing functions that could be useful in this regard [20]. Finally, studying tighter combinations of analysis steps and event-based visualization (e.g., to detect events on temporal data abstractions) could result in new powerful means for the visual analysis of time-oriented data.

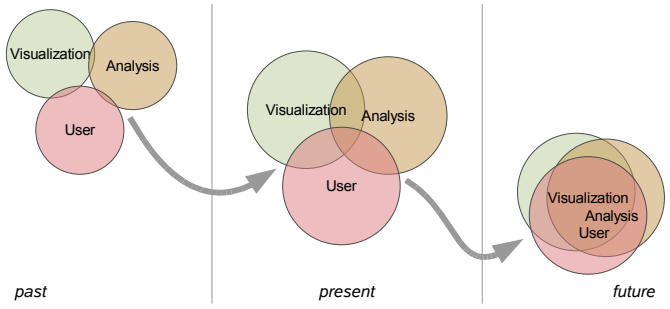

Fig. 17. To further advance a visually driven analysis of time-oriented data, it is necessary to integrate visual, analytical, and user-centered methods more tightly.

As a conclusion of our paper we would like to take a look at Fig. 17. Each distinct research field shown in the figure has yielded many powerful approaches. With this paper we tried to make a point on a better integration of visual, analytical, and user-centered methods. We suggest that these aspects are further advanced in a direction that leads to convergence of user-centered, visually driven analysis methods for time-oriented data.

ACKNOWLEDGMENTS

We would like to thank all people who have contributed to the many examples we present in this paper. The anonymous reviewers deserve special thanks for their valuable comments. The work presented here was partially supported by the DFG (German Research Foundation) and by the "Fonds zur Förderung der wissenschaftlichen Forschung - FWF" (Austrian Science Fund), grant P15467-INF.

REFERENCES

[1] B. Shneiderman, "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations," in *Proc. of the IEEE Symp. on Visual Languages*. IEEE Press, 1996, pp. 336–343.

[2] J. J. Thomas and K. A. Cook, "A Visual Analytics Agenda," *IEEE Computer Graphics and Applications*, vol. 26, no. 1, pp. 10–13, 2006.

[3] E. Hajnicz, *Time Structures: Formal Description and Algorithmic Representation*, ser. Lecture Notes in Computer Science. Berlin: Springer-Verlag, 1996, no. 1047.

[4] A. U. Frank, "Different Types of "Times" in GIS," in *Spatial and Temporal Reasoning in Geographic Information Systems*, M. J. Egenhofer and R. G. Golledge, Eds. New York: Oxford University Press, 1998.

[5] W. Aigner, "Visualization of Time and Time-Oriented Information: Challenges and Conceptual Design," Ph.D. dissertation, Vienna University of Technology, 2006.

[6] I. A. Goralwalla, M. T. Özsu, and D. Szafron, "An Object-Oriented Framework for Temporal Data Models," in *Temporal Databases: Research and Practice*, E. et al., Ed. Springer, 1998, pp. 1–35.

[7] W. Müller and H. Schumann, "Visualization Methods for Time-dependent Data - an Overview," in *Proc. of Winter Simulation 2003*, New Orleans, USA, Dec. 2003.

[8] S. F. Silva and T. Catarci, "Visualization of Linear Time-Oriented Data: a Survey (Extended version)," *Journal of Applied System Studies*, vol. 3, no. 2, 2002.

[9] M. Weber, M. Alexa, and W. Müller, "Visualizing Time-Series on Spirals," in *Proc. of the IEEE Symp. on Information Visualization 2001 (InfoVis01)*, Oct. 2001, pp. 7–14.

[10] J. V. Carlis and J. A. Konstan, "Interactive Visualization of Serial Periodic Data," in *Proc. of Symposium on User Interface Software and Technology (UIST)*, 1998.

[11] K. P. Hewagamage, M. Hirakawa, and T. Ichikawa, "Interactive Visualization of Spatiotemporal Patterns Using Spirals on a Geographical Map," in *Proceedings of Symposium on Visual Languages (VL)*, Tokyo, Japan, 1999.

[12] C. Tominski, J. Abello, and H. Schumann, "Axes-Based Visualizations with Radial Layouts," in *Proc. of ACM Symp. on Applied Computing*. ACM Press, 2004, pp. 1242–1247.

[13] ——, "Interactive Poster: 3D Axes-Based Visualizations for Time Series Data," in *Poster Compendium of IEEE Symp. on Information Visualization (InfoVis'05)*, Minneapolis, USA, 2005.

[14] W. Aigner, S. Miksch, B. Thurnher, and S. Biffl, "PlanningLines: Novel Glyphs for Representing Temporal Uncertainties and their Evaluation," in *Proc. of the 9th Intl. Conf. on Information Visualisation (IV05)*. IEEE Press, 2005.

[15] C. Plaisant, B. Milash, A. Rose, S. Widoff, and B. Shneiderman, "LifeLines: Visualizing Personal Histories," in *CHI '96: Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press, 1996.

[16] L. Chittaro and C. Combi, "Visualizing Queries on Databases of Temporal Histories: New Metaphors and their Evaluation," *Data and Knowledge Engineering*, vol. 44, no. 2, pp. 239–264, 2003.

[17] S. Havre, E. Hetzler, P. Whitney, and L. Nowell, "ThemeRiver: Visualizing Thematic Changes in Large Document Collections," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 1, pp. 9–20, 2002.

[18] R. L. Harris, *Information Graphics: A Comprehensive Illustrated Reference*. Oxford University Press, 1999.

[19] H. Hochheiser, "Interactive Graphical Querying of Time Series and Linear Sequence Data Sets," Ph.D. dissertation, University of Maryland, 2003.

[20] H. Doleisch, H. Hauser, M. Gasser, and R. Kosara, "Interactive Focus+Context Analysis of Large, Time-Dependent Flow Simulation Data," *Transactions of the Society for Modeling and Simulation International*, to appear 2007.

[21] J. Lin, E. Keogh, and S. Lonardi, "Visualizing and Discovering Non-Trivial Patterns in Large Time Series Databases," *Information Visualization*, vol. 4, no. 2, pp. 61–82, 2005.

[22] D. Keim, "Scaling Visual Analytics to Very Large Data Sets," Workshop on Visual Analytics, Darmstadt, June 2005.

[23] W. J. Clancey, "Heuristic Classification," *Artificial Intelligence*, vol. 27, pp. 289–350, 1985.

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

[25] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *Proc. ACM*

SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery. ACM Press, 2003.
[26] S. Miksch, W. Horn, C. Popow, and F. Paky, "Utilizing Temporal Data Abstraction for Data Validation and Therapy Planning for Artificially Ventilated Newborn Infants," *AI in Medicine*, vol. 8, no. 6, pp. 543–576, 1996.
[27] S. Miksch, A. Seyfang, W. Horn, and C. Popow, "Abstracting Steady Qualitative Descriptions over Time from Noisy, High-Frequency Data," in *Proc. of the Joint European Conf. on AI in Medicine and Med. Decision Making (AIMDM'99)*. Springer, Berlin, 1999, pp. 281–290.
[28] R. Bade, S. Schlechtweg, and S. Miksch, "Connecting Time-oriented Data and Information to a Coherent Interactive Visualization," in *Proc. of the 2004 Conf. on Human Factors in Computing Systems (CHI04)*. ACM Press, 2004, pp. 105–112.
[29] J. Lin, E. Keogh, L. Wei, and S. Lonardi, "Experiencing SAX: a Novel Symbolic Representation of Time Series," *Data Mining and Knowledge Discovery*, 2007, to appear.
[30] I. T. Jolliffe, *Principal Component Analysis*, 2nd ed., ser. Springer Series in Statistics. Springer Verlag, New York, 2002.
[31] S. dos Santos and K. Brodlie, "Gaining understanding of multivariate and multidimensional data through visualization," *Computers & Graphics*, vol. 28, pp. 311–325, 2004.
[32] S. Havre, E. Hetzler, and L. Nowell, "ThemeRiver: Visualizing Theme Changes Over Time," in *Proc. IEEE Symp. on Information Visualization (InfoVis'00)*, Salt Lake City, USA, Oct. 2000, pp. 115–123.
[33] T. Nocke, H. Schumann, and U. Böhm, "Methods for the Visualization of Clustered Climate Data," *Computational Statistics*, vol. 19, no. 1, pp. 75–94, 2004.
[34] W. Müller, T. Nocke, and H. Schumann, "Enhancing the Visualization Process with Principal Component Analysis to Support the Exploration of Trends," in *Proc. of APVIS'06*, 2006.
[35] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.
[36] J. J. van Wijk and E. R. van Selow, "Cluster and Calendar Based Visualization of Time Series Data," in *Proc. of the IEEE Symp. on Information Visualization 1999 (InfoVis'99)*, 1999, pp. 4–9.
[37] T. Nocke, H. Schumann, U. Böhm, and M. Flechsig, "Information Visualization Supporting Modeling and Evaluation Tasks for Climate Models," in *Proc. of Winter Simulation 2003*, New Orleans, USA, Dec. 2003.
[38] J. Seo and B. Shneiderman, "A Rank-by-Feature Framework for Interactive Exploration of Multidimensional Data," *Information Visualization*, vol. 4, no. 2, pp. 99–113, 2005.
[39] E. Keogh, H. Hochheiser, and B. Shneiderman, "An Augmented Visual Query Mechanism for Finding Patterns in Time Series Data," in *Proc. Fifth International Conference on Flexible Query Answering Systems*. Springer-Verlag, 2002.
[40] K. Henriksen, J. Sporring, and K. Hornbaek, "Virtual Trackballs Revisited," *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, no. 2, pp. 206–216, 2004.
[41] C. Tominski, "Event-Based Visualization for User-Centered Visual Analysis," Ph.D. dissertation, University of Rostock, 2006.
[42] S. dos Santos and K. Brodlie, "Gaining understanding of multivariate and multidimensional data through visualization," *Computers & Graphics*, vol. 28, no. 3, pp. 311–325, 2004.
[43] R. Sadri, C. Zaniolo, A. Zarkesh, and J. Adibi, "Expressing and Optimizing Sequence Queries in Database Systems," *ACM Transactions on Database Systems*, vol. 29, no. 2, pp. 282–318, 2004.
[44] D. H. House, A. S. Bair, and C. Ware, "An Approach to the Perceptual Optimization of Complex Visualizations," *IEEE Transactions on Visulaization and Computer Graphics*, vol. 12, no. 4, pp. 509–521, 2006.

Wolfgang Aigner is scientific researcher at the Department of Information and Knowledge Engineering (ike), Danube University Krems, Austria and lecturer at Vienna University of Technology, Austria. He received his PhD in computer science from Vienna University of Technology in March 2006 for his work on "Visualization of Time and Time-Oriented Information: Challenges and Conceptual Design". His main research interests include Visual Analytics, Information Visualization, Human-Computer Interaction (HCI), and User Centered Design.

Silvia Miksch is University Professor (Univ.-Prof.) and head of the Department of Information and Knowledge Engineering (ike) at the Danube University Krems, Austria. Since 1998 she is head of the Information and Knowledge Engineering research group (IEG), Institute of Software Technology and Interactive Systems (ISIS), Vienna University of Technology. Her main research interests include Information Visualization and Visual Analytics (in particular Focus&Context and Interaction techniques), Plan Management, and Evaluation of Knowledge-Based Systems in Real-World Environments (Health Care).

Wolfgang Müller is a professor for Media Education and Visualization and Vice Director of the Center for Learning with Digital Media at University of Education Weingarten. He received his PhD in computer science from Darmstadt University of Technology in 1999. His main research interests include E-Learning, Visual Analytics, Information Visualization, Interactive Storytelling, and Human-Computer Interaction.

Heidrun Schumann graduated at the University of Rostock (1977 Master degree, 1981 PhD, 1989 postdoctoral lecture qualification). Since 1992 she is heading the Computer Graphics Research Group at the Institute for Computer Science at the University of Rostock. Her research profile covers Information Visualization and Visual Data Mining, Mobile Interfaces, Rendering, and Image Presentation. She was heading the cross-institutional research group "MoVi - Visualization of Multimedia Information on Mobile Computer Systems", supported by the German Research Foundation (DFG). Her current research projects, supported by research institutions and industry, include development of scalable frameworks for information visualization, visualization of climate data, as well as mobile interfaces and user defined image transmission.

Christian Tominski received his diploma (MCS) from the University of Rostock in 2002. In 2006 he received doctoral degree (Dr.-Ing.) from the same university. Currently, Christian is working in a post-doctoral position at the Institute for Computer Science at the University of Rostock. He is cooperating with DIMACS at Rutgers University and the Fraunhofer Institute for Computer Graphics. Christian has authored and co-authored several articles all related to the visual communication of information. His main interests concern visualization of multivariate data in time and space, visualization of graph structures, and visualization on mobile devices. In his research, a special focus is set on highly interactive systems.