Event Tunnel: Exploring Event-Driven Business Processes

Martin Suntinger and Hannes Obweger ■ *Senactive Inc.*

Josef Schiefer and M. Eduard Gröller ■ *Vienna University of Technology*

Event-based systems monitor business processes in real time. The event-tunnel visualization sees the stream of events captured from such systems as a cylindrical tunnel. The tunnel allows for back-tracing business incidents and exploring event patterns' root causes. The authors couple this visualization with tools that let users search for relevant events within a data repository.

Today's networked business environments require adaptive and easily integrated systems. Event-based systems control business processes with loosely coupled systems. They enable the processing of large amounts of events, using them to monitor, steer, and optimize business processes with minimal latency. Typical application areas require fast decision cycles based on a large number of observable business events that can be used to discover exceptional situations or business opportunities. Typically, these areas include financial market analysis, trading, security, fraud detection, customer relationship management, and logistics such as tracking shipments and compliance checks.

The success of event-driven business solutions depends on ongoing learning. It's an iterative cycle, including the analysis and interpretation of past processing results and the conversion of these results into event-processing logic. Analysis tools tailored to the characteristics of event data answer questions such as

- Where did irregularities occur in my business?
- Did processes change over time?
- Upon which data were past automated decisions made?
- Did errors occur in the automated decision process?
- What happened at a certain point in time at a certain location, and who was involved?

To answer these questions, business analysts must have extensive retrieval tools to extract required data sets. They also need expressive visualizations to navigate through event data and recognize recurring patterns and irregularities that influence the business performance. We developed the Event Tunnel system to address these needs.

Data integration and retrieval

Continuous capturing and processing of events produces vast amounts of data. Efficient mass storage is required to store all events and prepare the data for later retrieval and analysis—in the following we refer to this storage as the *event space*. During event processing in our event-based system, an auditing service captures and stores events in the event space. Furthermore, the Event Tunnel's back-end system indexes events for quickly retrieving correlated events, and it precalculates metrics. (For further information on other event-based systems, see the "Related Work" sidebar.) Figure 1 (page 48) illustrates the overall architecture and data-integration process.

For maintaining information about business activities, events capture attributes about the context when the event occurred. For example, a typical order event could have the following attributes as context information:

- customer name (string),
- order ID (string),
- product ID (string), and
- price (numeric).

This attribute template defines the structure of a certain class of events and is called an *event type*. It indicates the underlying type of state change in a business process that the event reflects.

 Published by the IEEE Computer Society

Related Work

Event-based systems and applications are starting to be employed in industrial settings. However, event data-specific visualization and analysis tools are still in their infancy. In the business intelligence domain, many approaches exist, from online analytical processing (OLAP) to data-mining techniques. These technologies are applicable for the analysis of event-based operational business data as well, but they don't consider the nature and special characteristics of events. Compared to OLAP analysis,[1] our approach consciously omits an abstraction of the data into a set of multidimensional key figures and focuses instead on the events as the data points for the visual data representations. Visual patterns and clusters provide high-level views of the data, and drill-down operations are possible down to the level of a single event.

Ming Hao and colleagues propose VisImpact to reduce the complexity of business data by extracting impact factors that identify either single nodes or groups of nodes from business flow diagrams that influence business operations.[2] VisImpact finds relationships among the most important impact factors and supports an immediate identification of anomalies. Our approach doesn't depend on process models, and it visualizes process-behavior patterns based on events generated from IT systems. It doesn't reduce the complexity of the data but allows the investigation of causal dependencies of events in a business environment. By extracting and visualizing manageable data sets, users can discover (hidden) relationships between events as well as impact factors for business operations.

The event-tunnel visualization displays search results from event-data queries. Szabolcs Rosznyai and colleagues proposed the search engine Event Cloud to analyze business events based on event queries.[3] The system lets users search in large sets of historical events and uses a text-based view to display the search results. This is supported by the work of Marc Sebrechts and his colleagues, who showed that locating a search target is fastest in text-based views.[4] We argue that for event data a visual representation of query results is more valuable because multiple data dimensions can be encoded at once in graphical features of the rendering and interrelations of retrieved events can be displayed as well.

Elke Rundensteiner and her colleagues implemented the XmdvTool for multivariate data visualization, letting users view data from different perspectives.[5] We extended this approach by a visualization technique and show events without further abstraction in the context of their occurrence. Condition-based mappings enable the encoding of various event-data aspects in the rendering.

Polaris, proposed by Chris Stolte and his colleagues provides extensive visualization opportunities for multidimensional data, including glyph-based visual mapping to shape, size, orientation, and color.[6] We have adopted several of these techniques and propose a novel way to incorporate the time dimension into the visualization.

In information visualization, different approaches exist to display the time dimension in the data. John Carlis and Joseph Konstan[7] and Marc Weber and his colleagues[8] propose a time spiral, aiming at temporal patterns in periodic data. Arranging data in a spiral, their approaches provide the user with easy visual cues to both serial and periodic aspects of the data along with interactions such as the change in period over time. Ka-Ping Yee and his colleagues show a visualization approach that uses a radial tree layout method for supporting the interactive exploration of graphs.[9] In our approach, we combine and extend these techniques for displaying networks of correlated events that occur over a certain period of time.

References

1. E. Codd, S. Codd, and C. Salley, "Providing OLAP to User-Analysts: An IT Mandate," white paper, E.F. Codd & Associates, 1993.
2. M.C. Hao et al., "Business Process Impact Visualization and Anomaly Detection," *Information Visualization*, vol. 5, no. 1, 2006, pp. 15–27.
3. S. Rozsnyai et al., "Event Cloud-Searching for Correlated Business Events," *Proc. 4th IEEE Intl Conf. Enterprise Computing, E-Commerce, and E-Services* (CEC/EEE 07), IEEE Press, 2007, pp. 409–420.
4. M.M. Sebrechts et al., "Visualization of Search Results: A Comparative Evaluation of Text, 2D, and 3D Interfaces," *Proc. 22nd Annual Int'l ACM SIGIR Conf. Research and Development in Information Retrieval* (SIGIR 99), ACM Press, 1999, pp. 3–10.
5. E.A. Rundensteiner et al., "XmdvTool: Visual Interactive Data Exploration and Trend Discovery of High-Dimensional Data Sets," *Proc. 2002 ACM SIGMOD Int'l Conf. Management of Data* (SIGMOD 02), ACM Press, 2002, p. 631.
6. C. Stolte, D. Tang, and P. Hanrahan, "Polaris: A System for Query, Analysis, and Visualization of Multidimensional Relational Databases," *IEEE Trans. Visualization and Computer Graphics*, vol. 8, no. 1, 2002, pp. 52–65.
7. J.V. Carlis and J.A. Konstan, "Interactive Visualization of Serial Periodic Data," *Proc. 11th Annual ACM Symp. User Interface Software and Technology* (UIST 98), ACM Press, 1998, pp. 29–38.
8. M. Weber, M. Alexa, and W. Müller, "Visualizing Time-Series on Spirals," *Proc. IEEE Symp. Info. Visualization 2001* (INFOVIS 01), IEEE CS Press, 2001, pp. 7–14.
9. K.-P. Yee et al., "Animated Exploration of Dynamic Graphs with Radial Layout," *Proc. IEEE Symp. Info. Visualization 2001* (INFOVIS 01), IEEE CS Press, 2001, pp. 43–51.

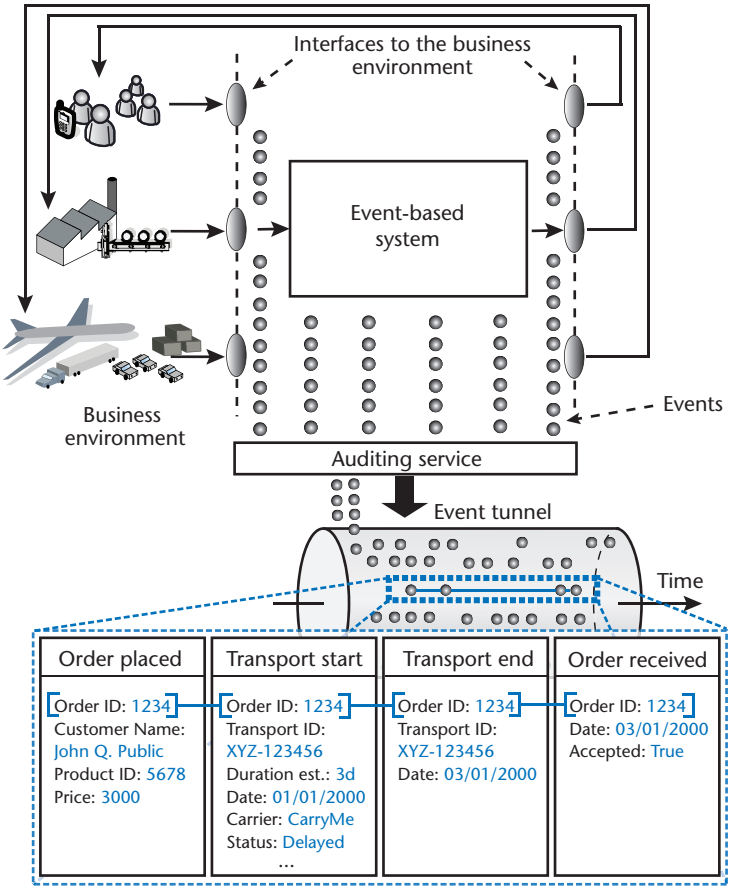


Figure 1. The system architecture. The event-space back-end system contains processed events for analysis and retrieval.

To back-trace and analyze business-process events, it's important to correlate temporally and semantically related events. We can use elements of an event context to define a relationship between events. Chains of correlated events reflect instances of a business process.

The event-tunnel visualization follows a query-driven approach. We unified the access to event data by specifying event filters and patterns with an event-access (EA) expression language. This language lets event attributes be accessed easily and supports modeling complex conditions, including calculations and aggregations. A typical search query is a set of conditions used to filter the events from the event space. A simple example of an EA expression is *Order.Price BETWEEN 100 AND 200 AND OrderDelivered.Status = 'Delayed'*. As the example shows, the query model lets the user define a search scope for retrieving correlated events that match a set of defined conditions. The Event Tunnel system applies the query model for multiple purposes. Conditions separate the event data into (eventually overlapping) groups. The system can then visually map condition-based coherences to color, shape, spatial proximity, and other visualization characteristics.

The event-tunnel visualization

Our event-tunnel visualization technique produces interactive, visual depictions of event-space data. The visualization is based on the metaphor of seeing the past stream of events as a 3D cylinder and providing two views of this cylinder: a side view plotting the events in temporal order (see Figure 2a) and a top view that looks into the stream of events along the time axis (see Figure 2b).

The event-tunnel top view maps the 3D view into the event-stream cylinder to a radial 2D rendering. Events in the tunnel's inner circles are displayed smaller to simulate perspective projection. The motivation behind this technique is an observation we made early in the planning phase: The entropy and business value of the most recent events are usually higher in comparison to similar but older events. This tendency results from the fact that the most recent events are more relevant for a business analyst and that the final events of a business process best characterize the outcome of a business case. The perspective projection in the event-tunnel top view reflects this requirement. The latest events are visible and appear larger at the outer rings.

Following the metaphor of an event-stream cylinder, one axis is determined and occupied unambiguously by time, whereas the remaining axis is assignable by a placement policy. During the course of our work, user feedback showed that integrating traditional, well-known scatter charts would be a valuable extension alongside the time-focused event-tunnel visualization we had until then. We therefore decided to generalize the approach and let users assign different placement strategies on both axes. In this way, the side view turns to a freely configurable scatter chart (see Figure 2c), while the top view becomes a radial layout chart with perspective projection of events (see Figure 2d).

Placement policies

As previously mentioned, both axes of the event-tunnel views are freely assignable. It's up to the users to apply suitable placement strategies. The effectiveness of such strategies depends on the users' objectives. Possible objectives might be to

- avoid overlapping events,
- display correlated events close to each other,
- emphasize sequences of events to recognize process patterns, or
- display distributions of attribute values.

It's hard to find a single strategy that fits all requirements at the same time. However, it's possible to define specialized policies that concentrate on certain aspects and try to optimize the output

with respect to these aspects. Ultimately, users can switch between placement strategies to find the output most effective for their purposes. In the following, we propose several placement policies and discuss their results.

EA expressions and time placement policy. The EA expressions placement policy applies an EA expression on each event and maps the result to the position on the event-tunnel view axis. The EA expression must be formulated to return a numeric value. The simplest EA expressions select attribute values directly, such as *Order.Price*. Complex expressions allow arbitrary computations on event attributes and map the result to the position of the event on the axis. In the following example, the placement is based on a computation considering three event attributes: *(Order.ProductPrice*Order.Quantity-Order.ShipmentCosts)/1.5*.

We can extract an event's occurrence time with an EA expression as well. We provide the time as a separate placement policy to the user because it's more intuitive to select and can be used as the default policy for the *x*-axis.

Sector placement policy. The sector placement policy focuses on distributions of events. It places them in nonoverlapping sectors on the axis. Applied to the top view's *y*-axis, the outcome resembles well-known pie charts. Conditions define sector memberships, although restrictions must be introduced to avoid multiset memberships.

The sector placement policy shows expressive results on medium to large event sets ranging from several hundred to 40,000 events. Combined with EA expressions or time placement policy accumulations, leaks and outliers become visible.

Correlations placement policy. This policy places correlated events on the same axis position. Combined with the time placement policy in the side view, the outcome closely resembles process charts such as GANTT diagrams.

Events can be part of several correlations. We address this by duplicating such events and placing a separate representation in each correlation chain to provide a definite outcome.

Centric event-sequence placement policy (CESPP). This policy extends the correlations placement policy. It avoids overlapping events and thereby gains additional expressiveness as characteristic patterns emerge from the placement. Originally, we designed the CESPP exclusively for the top view, but it can be applied in the side view as well.

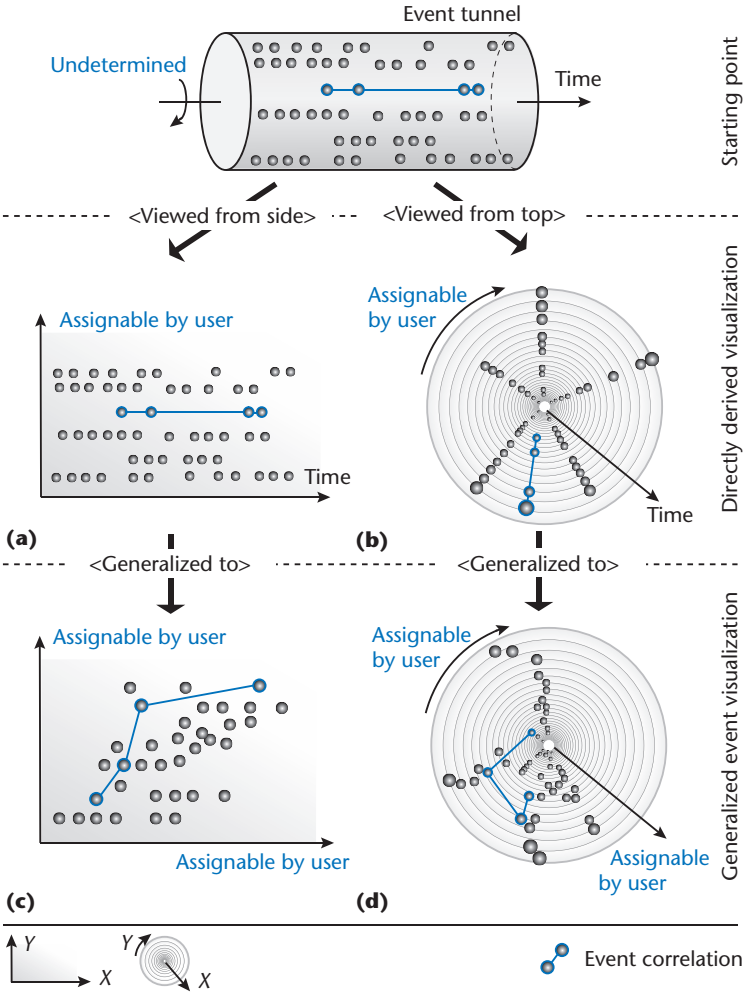


Figure 2. The event-tunnel visualization metaphor. (a) Side view of the event stream; *x*-axis is occupied by the time, *y*-axis is freely assignable. (b) Top view into the stream of events. Events are perspectively projected, and the axis around the tunnel is freely assignable. (c) Generalization of the side view to configurable scatter chart with both axes assignable by user. (d) Generalization of top view to configurable radial layout chart.

The algorithm in Figure 3 (next page) avoids overlapping events by orthogonal shifts. If necessary, the CESPP moves an event in the positive direction of the axis so that it doesn't overlap its predecessor. The algorithm converts specific patterns into abstract representations of underlying business-process instances. As regular instances of a business process proceed more or less according to a template, characteristic patterns emerge. This enables analysts to derive fuzzy, visual template representations for certain business processes. We can then assess anomalies by comparing the resulting pattern with the expected template. Therefore, the CESPP is well suited for analyzing smaller data sets to detect fine-grained causal relationships. Although most of the emerging patterns are application and data specific, we can characterize several

```
CESPP(events, axis)
comment : Plot events according to the CESPP
for i ← 0 to events.length
do
    if i = 0
      then PlotEvent(events[i])
    else
        events[i].Position[axis] ← events[i−1].Position[axis]
        if Overlap(events[i],events[i−1])
          then Shift(events[i])
        PlotEvent(events[i])
```

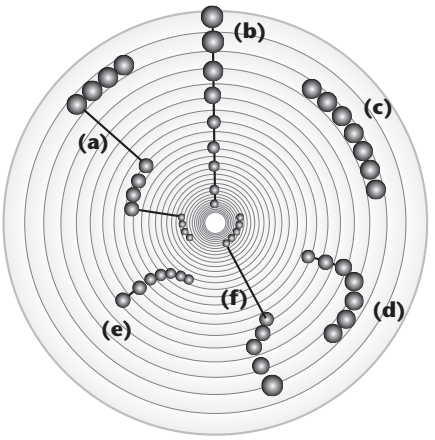


Figure 3. Examples of characteristic business-process patterns in a combination of the centric event-sequence placement and time placement policies. Table 1 explains the individual patterns.

basic, high-level patterns. Figure 3 schematically shows these patterns for the combination of the CESPP and the time placement policy. Table 1 lists how these patterns can be interpreted.

Fill-space placement policy. This policy stacks events that would overlap on top of each other. In combination with the EA expression or time placement policy, the space-filling mechanism shows distributions of events similar to a histogram. Because it neither requires user configuration nor depends on event correlations, it's well suited as the side view's initial layout.

Combining placement policies. Our model foresees the combination of the placement policies mentioned previously. In Figure 4, we provide a schematic illustration of possible combinations. Keep in mind that certain limitations exist. To generate reasonable results, CESPP, correlations, and fill-space placement policies need to be combined with a placement policy that performs a definite placement for each single event.

Mapping data to event glyphs

Placement policies map data attributes or correlation group membership to an event's position in the visualization. In addition, an event's appearance can encode further information. Events are represented as glyphs consisting of an inner surface surrounded by a border (see Figure 5). These elements' shape, size, and color encode event attributes. Coding the event glyphs' size parameters is suitable for the mapping of (nearly) continuous attributes. For shape and color coding, we follow a condition-based approach. Although this provides great flexibility, you have to deal with overlapping conditions. For color coding, Event Tunnel divides the surface into equal-sized colored sectors Shape coding, however, is limited. The system can't render multiple shapes in a straightforward fashion. Each event glyph has exactly one shape.

Table 1. Interpretation of business-process patterns in Figure 3.

Pattern	Interpretation
Stair pattern (a)	Business process characterized by several idle times (potentially exceptional delays).
Noninterfering chain (b)	Long-running process in which stages pass straightforwardly in regular time steps.
Parallel chain (c)	Fast-executing process (automated or machine controlled) without idle times.
Acceleration worm (d)	Process that accelerates continuously.
Deceleration worm (e)	Process that decelerates continuously.
Rattlesnake (f)	Reflects one extreme delay in the execution of a process.

Data aggregation

The visualization approaches presented so far enable data analysis for single events. The necessity and usefulness for detailed business-process analyses and back-tracing of business incidents is obvious. However, hundreds of thousands of events might stream into the system, and even single sequences of correlated events might consist of too many events to be displayed independently. To address our approach's scalability and gain additional expressiveness, we propose aggregation facilities.

By aggregation, we mean the process of consolidating a set of events into a single aggregate data point. For instance, users can set an aggregation for order process correlations to consolidate each order process (represented by multiple events) to a single data point. After performing the aggregation, the user can map data attributes characterizing the group of consolidated events to the aggregate point's representation. For instance, our system can map process measures such as order delay, gross order costs and profit, and satisfaction status to the data point's color, size, or shape.

In other application examples, one correlation might exist per customer, which correlates all events of a certain customer into a sequence of interrelated events. In such cases, the event sequence

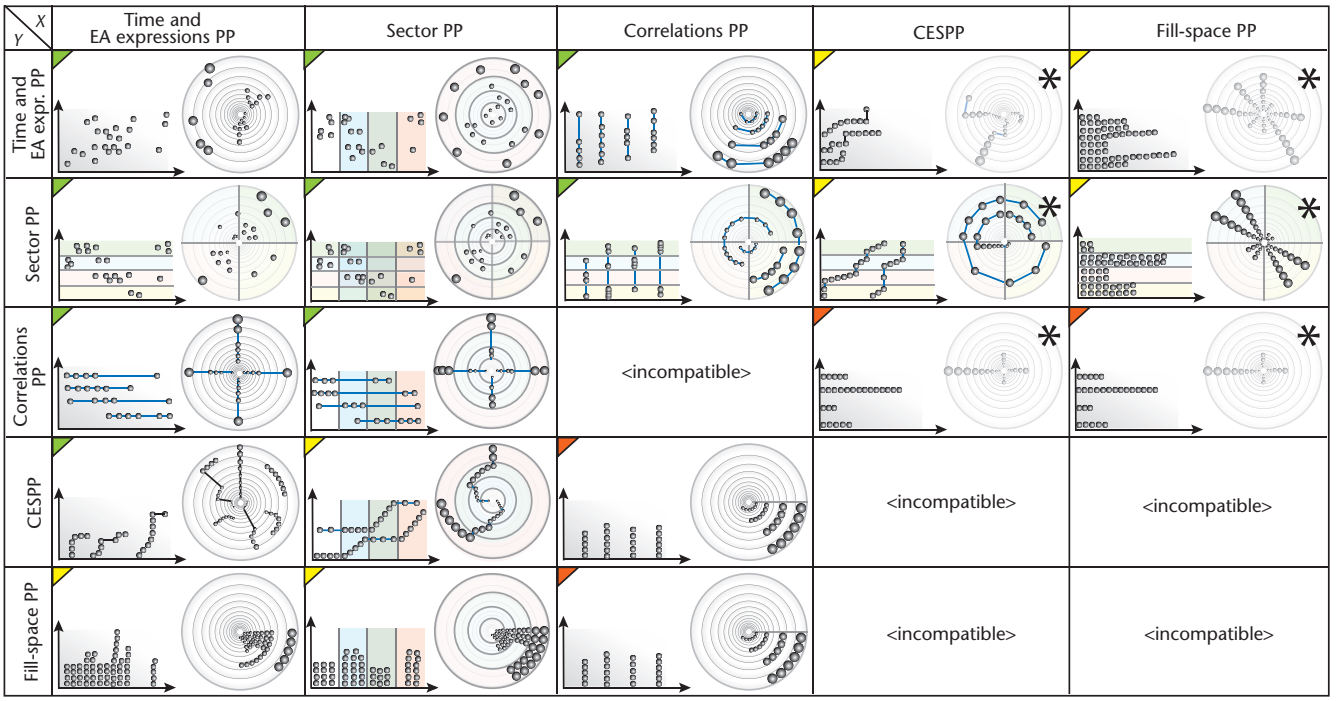


Figure 4. Placement policies (PPs) compatibility matrix. (CESPP stands for centric event-sequence PP.) This figure schematically sketches the visualizations that can be produced by combining the PPs, shows which PPs can be combined at all, and gives a hint on which combinations—based on experience—are useful.

might be aggregated to a customer data point, which allows visualization and comparisons of customers' key characteristics. The advantage of this approach is that analysts can drill down from an aggregate point to a single event. In the visualizations, an aggregate point can be collapsed or expanded. In the collapsed state, its representation resembles single-event glyphs. In the expanded state, the events contained in the cluster are plotted on a resizable circle. The metric determining the events' distance from the center of this circle is customizable.

The analysis framework

The event-tunnel top and side view visualizations generate interactive views on the event data. We embedded these views into a configurable workspace for event analysis and mining purposes. The workspace includes facilities for querying the event space, filtering, and mapping data to visual features in freely configurable worksheets. Each worksheet might contain several event data visualizations, such as the tunnel views. Other event data visualizations are a text-based view resembling the output of common Web search engines, a details view listing the attributes of events and correlations, and a data-table view similar to known spreadsheet applications.

The concept of worksheets integrates with the concept of filtering and data selection via EA expressions. Each tunnel view holds a data filter in the background, which selects the subset of data to be displayed. In this way, our system can modify a view's data set by updating the corresponding filter expression. It's possible to do this transparently for the user—for instance, when offering drag-and-drop operations to add data to a view.

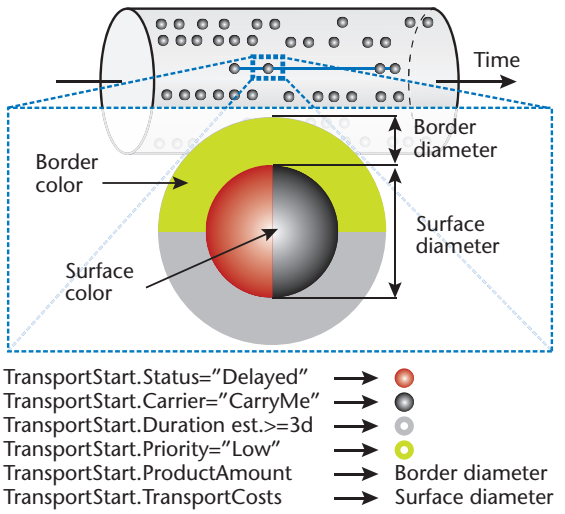


Figure 5. In the Event Tunnel, event attributes can be mapped to shape, colors, and sizes of event glyph parameters. This figure illustrates an event glyph. Event attributes are mapped to glyph parameters with conditions. In case of colors, if multiple conditions are met, the area is subdivided.

Figure 6 (next page) provides a screen shot of the analysis workspace. The user can add an arbitrary number of worksheets. The figure shows an example of a worksheet containing an event-tunnel top view, a side view, a text view, and a data-table view. The analysis framework targets business analysts and developers of business logic, which is based on company-specific event processing.

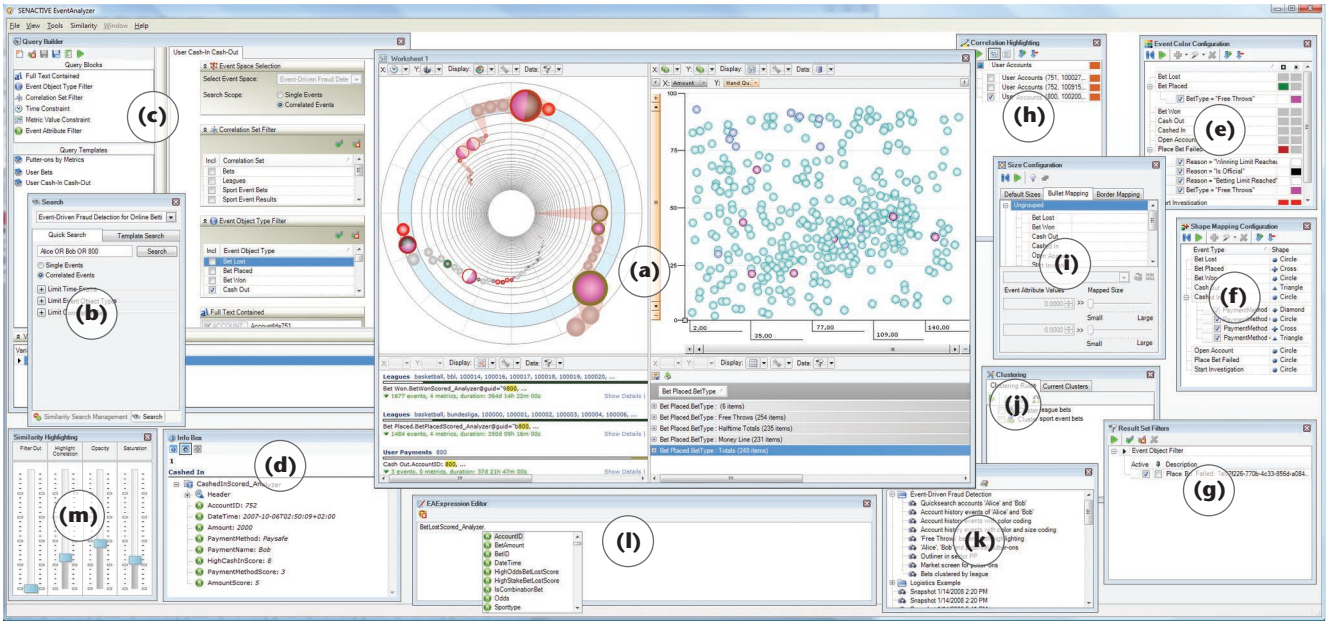


Figure 6. The analysis workspace. (a) Visualization worksheet, (b) quick-search editor, (c) graphical query builder, (d) text box, (e) color configuration panel, (f) shape-mapping panel, (g) filter manager, (h) correlation highlighting panel, (i) size mapping editor, (j) management console for aggregation, (k) snapshot management console, (l) EA expressions editor, and (m) similarity highlighting controller.

Knowledge discovery case study

To evaluate the visualization techniques previously presented, we applied the analysis framework for a business application in the fraud management domain. Fraud detection and prevention is a major issue in technology-driven business domains relying on online payment solutions and customer interactions. A market heavily affected by fraud cases recently is online betting and gambling. Various forms of fraud exist, ranging from physical attacks by hackers over money laundering to the abuse of insider information about sporting events.

Event-based systems help detect fraud and proactively prevent it in near real time by using rules to continuously evaluate customer interactions. This permits an automated intervention in case of suspicious activities. Implementing an efficient fraud detection and prevention event-based system requires exhaustive knowledge of the underlying business processes and reoccurring fraud patterns. Keeping such systems up to date requires data tracing, anomalies discovery, and pattern characterization.

Data tracing is a stepwise reproduction of recent, influential occurrences. Typically, this step is triggered by some clue that serves as the starting point for further investigation efforts. Our framework's query-driven visualization approach complies well with data-tracing tasks. Analysts translate the initial clue into a query to extract a limited data set for the visualizations. They can then subsequently narrow or broaden the search scope to the required granularity level.

For anomalies discovery, our approach is currently limited to a manual anomaly-discovery process supported by the visual accentuation of data characteristics with the application of placement policies, glyph mapping, and highlighting techniques.

Making the collected information useful after these analysis steps requires generalizing and characterizing the discovered patterns. Pattern characterization includes influential factors (such as key figures and threshold levels) as well as behavioral patterns in event sequences whose combination makes up a reference pattern. Event-based systems use this reference pattern for discovering similar cases.

We used our event-tunnel analysis framework to attain this knowledge and evaluated it with an event-based fraud detection and prevention system for online betting providers. We generated the data in the following examples using a simulation model that comprises known user-behavior patterns in the simulated events. For the evaluation we simulated ordinary bet placing, cash-in and cash-out events, and randomly added 1 to 2 percent fraud cases from several available fraud templates. We parameterized the templates to vary in structure and conspicuity.

Tracing customer activities

In a requirements study for an online-betting provider, we observed that a complete department of security analysts was concerned with data tracing. For fraud investigations, security analysts had to regularly back-trace customer actions in case of system alerts. The current practice mainly relies

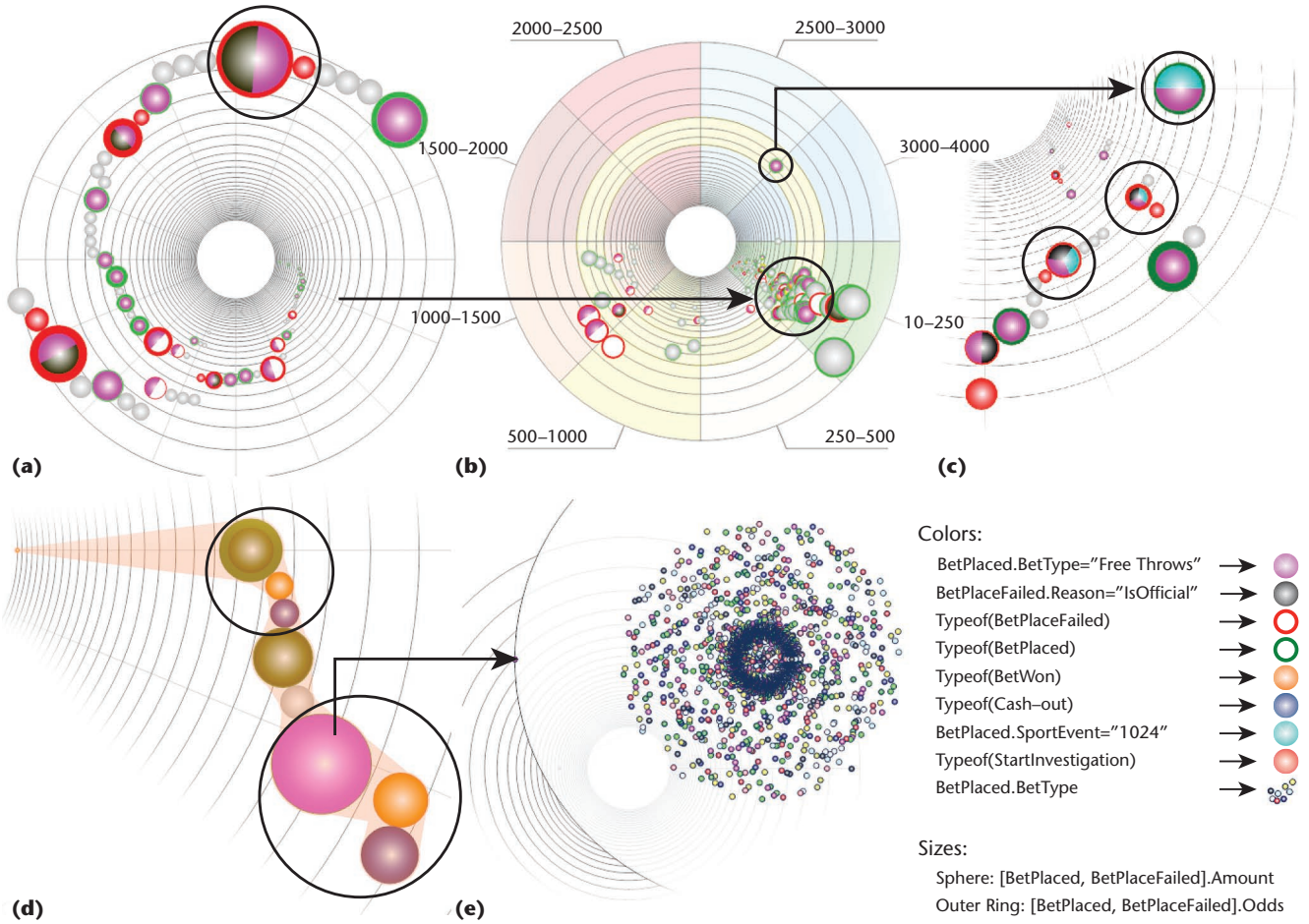


Figure 7. Example of a stepwise investigation of online-betting data. (a) Account histories of two users, (b) bet amount distribution for a selected sport event (the ranges of numbers represent the distribution's sector labeling), (c) suspicious occurrences at a certain point in time, (d) a putter-on account profile, and (e) cluster of high-stake bets with an outlier.

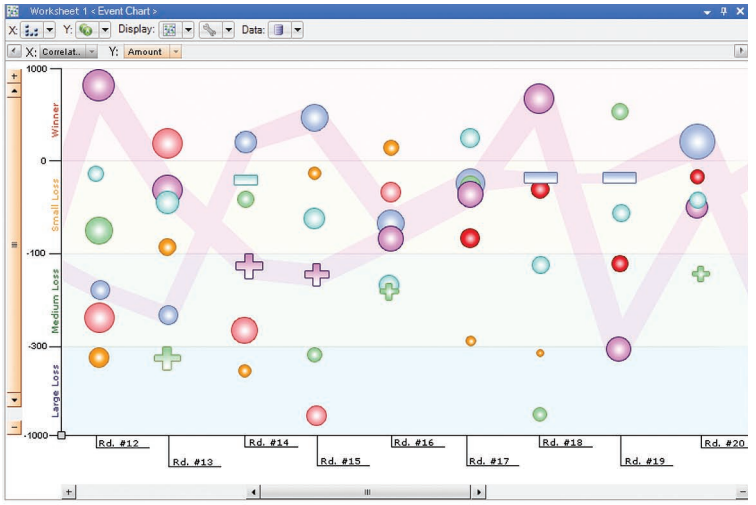
on screening log files, from which experienced analysts derive a mental picture of recent occurrences and customer behavior. Fraud investigations target single users, groups of users, or the analysis of complete markets and leagues. In the following, we assume that an event-driven fraud-detection system automatically generates alerts for users with suspicious behavior. The security analysts regularly receive clues (the account IDs of users with fraudulent behavior) that serve as starting points for a data-tracing-analysis step.

In Figure 7, we show an excerpt of a stepwise tracing and discovery process. We based it on an illustrative case that a test user discovered in the simulated data set. Figure 7a shows the historical events of two user accounts. Both users triggered an alert for the same sporting event (see Figure 7b) and bet type. The plot shows that they failed to place a bet because the system recognized them as officials. You can see that the sequences of actions of both users strongly correlate. Mapping the event attribute bet type to color reveals that both users exclusively place "free throws" bets (see Figure 7c).

The example proves that the information density in the event tunnel speeds up the reconstruction of business cases. Event-tunnel visualizations are more efficient compared to text-based output because they display multidimensional information in one view. Although a text-based list of events provides one degree of freedom (the sequential order), the event-tunnel visualization provides a time-based display, two size dimensions (surface diameter and border diameter), several color dimensions, and shape. The advantage over a text-based view becomes more obvious when analyzing multiple account profiles: The exact temporal order of all users' events is reflected in the visualization, and temporal coherences are immediately visible.

Discovering anomalies in betting behavior

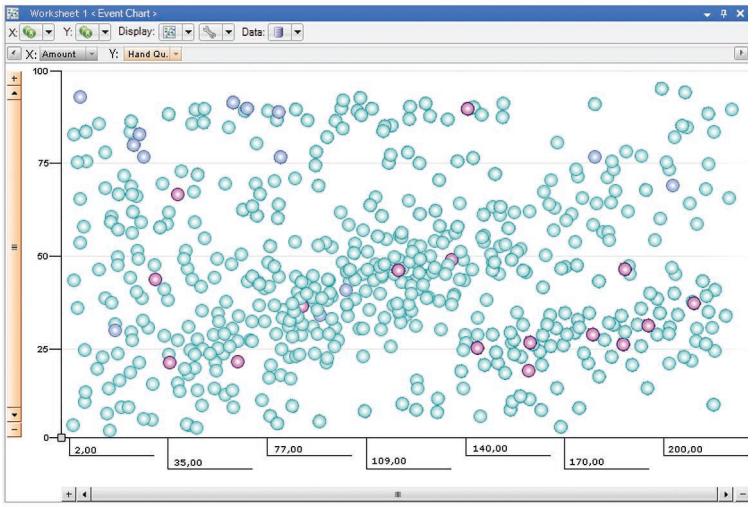
Data-tracing activities focus on a specific business entity and user and track the events related to this entity. A more general approach is to start at a higher granularity level and drill down to more detailed data. For instance, to investigate the bet type for which both bet placements failed in the previous



X-axis: Correlation per round *Y*-axis: HandPlayed.TotalAmount

HandPlayed.Behavior ="Normal"
HandPlayed.Behavior="Push"
HandPlayed.Behavior="Irrational Fold"
HandPlayed.Player.ID
HandPlayed.Player.Success
Buddies

(a)



X-axis: HandPlayed.Amount *Y*-axis: HandPlayed.CardsQuality

Suspicious player 1
Suspicious player 2
Other players

(b)

Figure 8. Sequence of poker hands played over several rounds with suspicious behavior highlighted by (a) shapes and (b) a scatterplot of hands played with the stake on the *x*-axis and the card quality on the *y*-axis.

example, analysts can broaden the search scope to all recent "free throws" bets in a certain market. Figure 7b shows a plot of bet events with the sector placement policy. The bets scatter around the tunnel according to the bet amount. The bet-placing failure events detected in the previous step appear in the bulk of average bet amounts. The plot exposes one salient outlier temporally related to the detected bet-placing failure events. This data point presents a valuable link to conspicuous data.

In Figure 7c, the account history events of the user who placed this bet are plotted in relation to the already detected suspicious account profiles. From the color and size mapping, analysts can conclude that the third user successfully placed a high-stake bet equivalent to the bet that failed for the two other users. One possible interpretation could be that user three is a *putter-on* who placed bets for people who are prohibited to bet, such as officials and players. The account profile of user three plotted in Figure 7d supports this hypothesis. The marked areas represent a recurring sequence of bet placing, bet won, and immediate cash-out. This sequence characterizes the putter-on pattern. The example shows that a single visual clue in a plot can lead analysts to immediately navigate to formerly unconsidered data. We found that in practice, such navigation chains can be followed by analysts over more than 10 steps before performing a completely new query. For example, the step in Figure 7d could lead to a further broadening of the search scope to investigate whether the system should have prohibited the high-stake bet of user three. Figure 7e plots recent high-stake bets in a cluster. The plot shows that one bet is an outlier. This could be a data error or failure of the event-processing system.

Applications in online poker

Within the gambling domain, online poker has grown to be one of the most popular online games during the past couple of years. With increasing popularity, the poker domain has also attracted a bulk of crooks. The main problem in online poker is collusion. Several people on a virtual poker table might be in contact, unrecognized by the other players, to take advantage of their collective knowledge. The recognition and prevention of collusion is challenging, requiring ever-evolving mechanisms to cope with continuously changing and improving strategies of fraudsters.

Another task for poker providers is proving collusion. The number of customer complaints on collusion is increasing drastically. As a result, providers must be able to trace data and find evidence for specific cases. To show how you can effectively assess collusion, we used our event-tunnel visualization to display interactions on a poker table over the course of several rounds of gameplay. Figure 8a shows a sequence of events for six players.

On the horizontal axis, we used the correlations placement policy. We correlated the events per round so the events of each round are plotted on the same *x*-position. On the *y*-axis, we used the sector placement policy to plot the gain or loss of each player per round. Distinct colors distinguish the players. In addition, during a preprocessing step we assessed suspicious behavior such as a fold with a very good hand (irrational fold) or pushing with a bad hand. Typically, irrational folding is a strong indicator of collusion if another player on the table has a strong hand. In many cases it means that the irrational folder knows of that

hand, which is the reason to fold. Such behavior is highlighted by shapes of the events.

Measures on customers, such as how often and how long two or more players play on the same table, can be used to calculate a buddy-score. We highlighted such buddies in the visualization with a colored correlation band. The glyph size is determined by the accumulated player success in the current session. From the plot, you can derive the following information: Two players are buddies (purple and light-blue glyphs). When the purple player showed pushing behavior, the light-blue player won. At least in one round the light-blue player showed irrational-folding behavior and the purple player won. Compared to the other players, these two players were more successful. Some other players showed pushing behavior as well but weren't successful (probably just a bluff).

In Figure 8b, the EA expressions placement policy occupies both axes. On the *x*-axis, the played amount of money is plotted while the quality of the cards is laid on the *y*-axis. The two identified buddies from the first plot are highlighted by colors. The scatterplot shows a slight correlation between the quality of the hand and the stake and an accumulation of high-stake games with poor hands, which might be bluffs and pushers. From the plot, we can determine the overall behavior of certain players among all other players.

Case study results

The visual sequence patterns resulting from placement strategies and glyph-based mapping to accentuate various data dimensions combined with query and filter facilities enabled users to continuously detect anchor points in their navigation chain through the data. With the ability to store color, shape, size, and placement policy configurations, tracing becomes more efficient because a visual representation of customer activities can be generated with a few clicks once a template configuration is available. Considering these advantages, the event-tunnel visualization outperforms the current practice of manual log-file screening. One tradeoff in the top view compared to classical methods like histograms or scatterplots is the difficulty of labeling. We tried to alleviate this problem by showing tool tips with event-data information and metrics to the user when hovering over an event.

We plan to further extend the Event Tunnel system in future projects. For example, condition-based operations (such as shape mapping, coloring, filtering, and querying) are limited to absolute matching of event-data attributes. Semantic similarity operations would enrich the power of these retrieval and highlighting mechanisms. The characterization of event-sequence patterns is one of the strengths of the event-tunnel visualization. Currently, we're elaborating on mechanisms to extract data that follow a given visual reference pattern as an extension to the current query engine.

Acknowledgments

The Austrian FFG Bridge project Sim-CT (812136-SCK/KUG) partially funded the work presented in this article. We'd also like to thank the development team at Senactive Inc. for its outstanding support throughout the implementation of the analysis framework.

Martin Suntinger *is a graduate student at the Institute of Computer Graphics and Algorithms, Vienna University of Technology, and product manager at Senactive Inc., where he is responsible for the development of analysis and visualization solutions. His research interests include information visualization, business intelligence, and data mining. Suntinger received his BS in computer science from the Vienna University of Technology. Contact him at msuntinger@senactive.com.*

Hannes Obweger *is a graduate student at the Institute of Computer Graphics and Algorithms, Vienna University of Technology, and a software architect at Senactive Inc. His research interests include information visualization, complex event processing, and data mining. Obweger received his BS in computer science from the Vienna University of Technology. Contact him at hobweger@senactive.com.*

Josef Schiefer *is a researcher at the Institute of Software Technology and Interactive Systems, Vienna University of Technology, and technical director at Senactive Inc. His research interests include complex event processing, business intelligence, and business-process management. Schiefer received his PhD in information systems from the University of Vienna. Contact him at js@ifs.tuwien.ac.at.*

M. Eduard Gröller *is an associate professor at the Vienna University of Technology, and adjunct professor of computer science at the University of Bergen. His research interests include computer graphics and flow, volume, medical, and information visualization. Gröller received his PhD in computer science from the Vienna University of Technology. He is a member of the IEEE Computer Society, ACM, GI (Gesellschaft für Informatik), and OCG (Austrian Computer Society). Contact him at groeller@cg.tuwien.ac.at.*