

# The Scalable Reasoning System: Lightweight Visualization for Distributed Analytics

William A. Pike<sup>1</sup>

Joe Bruce<sup>2</sup>

Bob Baddeley

Daniel Best

Lyndsey Franklin

Richard May

Douglas M. Rice

Rick Riensche

Katarina Younkin

Pacific Northwest National Laboratory

## ABSTRACT

A central challenge in visual analytics is the creation of accessible, widely distributable analysis applications that bring the benefits of visual discovery to as broad a user base as possible. Moreover, to support the role of visualization in the knowledge creation process, it is advantageous to allow users to describe the reasoning strategies they employ while interacting with analytic environments. We introduce an application suite called the Scalable Reasoning System (SRS), which provides web-based and mobile interfaces for visual analysis. The service-oriented analytic framework that underlies SRS provides a platform for deploying pervasive visual analytic environments across an enterprise. SRS represents a “lightweight” approach to visual analytics whereby thin client analytic applications can be rapidly deployed in a platform-agnostic fashion. Client applications support multiple coordinated views while giving analysts the ability to record evidence, assumptions, hypotheses and other reasoning artifacts. We describe the capabilities of SRS in the context of a real-world deployment at a regional law enforcement organization.

**KEYWORDS:** Web visualization, mobile visualization, analytic reasoning, law enforcement, multiple views

**INDEX TERMS:** H.5.3 [Information Systems]: Information Interfaces and Presentation—Group and Organization Interfaces; C.2.4 [Computer Systems Organization]: Computer-Communication Networks —Distributed Systems

## 1 INTRODUCTION

For advances in visual analytics to benefit as large a population of end users as possible, application designers must consider ways to make visualization accessible and ubiquitous. One mechanism to achieve these aims is to develop analytic solutions that are *scalable* in two key ways. First, tools must scale well to large numbers of potentially distributed users; in addition to simply supporting many synchronous users, scalability along this dimension requires that tools have low barriers to entry and integrate smoothly into existing workflows. Second, tools must scale across the range of interfaces in use within an enterprise; analysts and investigators may, in the course of their work, use handheld devices, desktop computers, and large collaborative displays. Rather than require the use of special purpose tools at each phase of their work, it is desirable to produce systems that offer the same underlying analytic capabilities but with interfaces customized to the affordances of each platform.

This paper describes our approach to creating and deploying a scalable visual analytics environment. The Scalable Reasoning

System (SRS) is a web service-based analytic toolkit that allows a library of interoperable, customizable components to be integrated into applications that bring the benefits of “thick client” standalone environments to distributed user bases. These components currently include such capabilities as data clustering, temporal trend identification, and geographic analysis. Our goal is to support an ecosystem of interoperable analytic services that allows tool developers, and eventually users, to create “mashups” that connect these services in flexible ways to meet changing analytic requirements. Interfaces to SRS are lightweight in that they offload computationally intensive processing to remote services and present simple, streamlined interfaces to their users.

The need for scalable analytic infrastructures is particularly acute in domains where the analysis function is distributed across large numbers of investigators, such as in law enforcement. We have been implementing SRS at a regional law enforcement information sharing network with more than 70 local, state, and federal member agencies and over 11,000 users. The diversity of user roles, requirements, and analytic strategies in this organization suggests solutions that are customizable and flexible to a range of analytic contexts, from desktop to field. While SRS is designed as a general-purpose analytic environment, we use examples from our experience deploying it at this organization to illustrate the capabilities of our approach.

A growing body of research recognizes the benefits of distributed web-based [1] and mobile [2, 3] visualization systems. Such systems bring visual analytic capabilities to communities that previously lacked them. For SRS to achieve this, we decouple the analytic algorithms that underlie a visualization from the rendering of that view. This approach allows any client application (indeed many different applications at once) to make use of these analytic services, meaning that any device that can connect to a service can make use of that service’s analytic capabilities. This separation of computation and visualization allows platforms that make up in ubiquity what they lack in computational power (such as handheld devices and web browsers) to reap the benefit of computationally intensive analytics. Analytic algorithms become platform-agnostic and client applications have the freedom to render analytic results in a manner appropriate to the affordances of their platform.

SRS also provides explicit support for the analytic reasoning process. Too often the insight generated through visual discovery is left tacit in the analyst’s mind or recorded in forms disconnected from exploratory tools. SRS clients can embed a graphical “reasoning whiteboard” on which users can link features discovered through exploratory visualization with reasoning structures such as emerging hypotheses.

Patterson et al. [4] identify a recursive analytic process for identifying high-value information that involves three states: explore, enrich, and exploit. SRS enables analysts to efficiently *explore* large data spaces to identify items of potential interest; through coordinated services and displays they *enrich* their information collections by weeding out lower-value items; high-

<sup>1</sup>william.pike@pnl.gov <sup>2</sup>joseph.bruce@pnl.gov

value material is then *exploited* to discover specific events, attributes, and relationships of interest.

## 2 A CASE STUDY IN LAW ENFORCEMENT

Increasing amounts of incident data are available to law enforcement personnel via regional and national information sharing systems. These systems, such as Nlets<sup>†</sup>, provide federated query capabilities that allow officers and analysts to aggregate information from across jurisdictions. However, with greater access to data comes greater data overload; queries can return hundreds and even thousands of results, making the task of identifying incidents, people, or patterns of interest a difficult one.

Currently, few analytic capabilities are available to users of these networks. Search results can sometimes be exported into systems such as Analyst's Notebook<sup>‡</sup>, but using graph visualization techniques as the point of entry to large incident data sets often results in the "spaghetti graph" problem that can make quick identification of key patterns difficult. In addition, it is desirable to be able to analyze data *in situ* (i.e., producing visualizations immediately as queries are run, allowing analysts to refine their queries). Further complicating analysis is the streaming nature of the data; incident records are often available in near real-time from providers such as police dispatch systems, meaning that users often want to understand the temporal evolution of their data, especially in time-critical situations such as incident response. In such circumstances manually exporting static data collections to standalone tools is inefficient.

Through formative interaction with law enforcement users, we have identified three SRS user categories:

- **Analysts** are "traditional" users of visual analytic technology; they typically work on longer-term, strategic analyses such as trend identification.
- **Investigators** typically work on shorter-term, tactical analyses such as individual crime cases. Their task is often to identify suspects and close cases.
- **Field officers** have the most immediate analytic needs; based on an observation or person of interest, they need the ability to quickly identify additional information that will help them determine a course of action.

Our approach is to generalize the core analytic functionality common to all three into abstract components, and implement interfaces to these components that incorporate the features required by each.

## 3 SERVICE-ORIENTED ANALYTICS

To bring lightweight analytic capabilities to distributed users with differing roles, SRS consists of a data store, which may be external to the system, a suite of web services that operate over this data, and a set of client interfaces that implement the web services. Componentizing an analytic application through services allows analysis algorithms to be swapped out easily.

### 3.1 Data model

SRS uses a flexible data model that allows incident information to enter its repository from multiple providers. There is no single schema to which all data to be analyzed by the system must conform. Instead, each data record is stored as an XML document whose schema file is available to the system; this schema is used to control how the data are parsed and transformed. Typically, records in the law enforcement implementation of SRS are represented in the Global Justice XML Data Model (GJXDM), although narrative text records can be represented as well.

<sup>†</sup> <http://www.nlets.org>

<sup>‡</sup> [http://www.i2inc.com/products/analysts\\_notebook/](http://www.i2inc.com/products/analysts_notebook/)

The central object in the SRS data model is a generic "Document" table. This table contains identifying metadata for a particular data record, and either the content of the incident record itself or a pointer to the external location of the incident record. Documents may also contain user-created data, such as annotations to source records.

### 3.2 Service layer

Analytic operations executed over data stored or referenced in the data layer are performed via web services. Any client application that is capable of calling a web service and reading/writing SOAP enveloped data may utilize the server-side capabilities of SRS.

At present, our service library consists of nearly 60 interoperable analytic components. These services provide control over most aspects of analytic application development, from creating data repositories and adding information to them, to executing complex queries, to specifying data processing steps and requesting visualizations. A tool developer can construct an analytic application by chaining these services together, or integrating them with other third-party services.

Services that perform analytic functions, described later in this section, output rendering rules that client applications interpret to produce a visual display. Rather than output complete graphics, which would require versions customized to each display environment, these services produce descriptions of a visual space that client application developers can decide how to render. Communication overhead is a key consideration in distributed systems where some users will be interacting through low-bandwidth mobile networks. Transmitting declarative representations rather than complete displays helps achieve parsimonious communication between client and server and reduces latency, because the client can often use these rules to handle some types of interaction (such as zooming and panning) without making additional requests of the server.

In the following sections we describe some of the analytic services available through SRS; Section 4 then introduces a client-side law enforcement application that implements these services.

### 3.3 Query

Queries are the basis for identifying record sets to analyze in SRS. SRS query services accept requests made in an XML query schema that represents not just the query conditions themselves, but additional information such as how often to re-run a query, or whether the query should be stored as a "filter" to alert the user to new incidents that meet these conditions. This query specification creates application independence and abstracts details such as where data are actually stored, the schema of those data stores, or what search engines might subsequently be called by this service to retrieve results.

### 3.4 Data clustering

One of the central requirements of our law enforcement users is the ability to detect associations between incident records across jurisdictions and over time. *Crime analysts*, for instance, often attempt to identify broad patterns in incident data to characterize the nature of crimes in an area. In the course of working on an individual crime case, *investigators* often expand their inquiry to determine if other, similar cases exist that might provide leads. Finally, *field officers* may perform a field interview and want to immediately know if the topics covered in that interview relate to other known crimes.

In each case, data clustering provides a mechanism for transforming the large amounts of incident data available to our users through queries into displays that help them answer these questions. A query to SRS returns a set of identifiers for records that satisfy this query. The client may choose to request these

records so that they may be displayed in a traditional list, or it may pass these identifiers to other services that will perform computation over them.

To request that these records be clustered into sets of potentially related incidents, the client submits their identifiers to a `CreateHarvest` web service. A harvest converts each record into a vector of top terms (or key-value pairs for structured records), modified from the approach used in IN-SPIRE [5]. A 200-dimensional space containing the terms that best discriminate between incidents is projected to two dimensions to create a renderable cluster space.

To reduce the amount of data that must be transmitted to the client to describe this cluster space, we apply a hierarchical clustering algorithm. This algorithm aggregates the records in the harvest into a specified number of clusters, which can either be determined by the system (typically, the square root of the number of incidents) or by the user (who may ask that a given result set be grouped into  $n$  clusters). Child clusters for any cluster in this set may be requested through subsequent web service calls, indefinitely until clusters contain only one document.

Figure 1 provides a sample response from `GetClusters`. The response includes a set of `ClusterInfo` elements corresponding to each cluster found in the source data. To overcome the stateless nature of web applications, we have developed a mechanism for uniquely identifying features in a visualization so that the client may communicate to the server what features a user is interacting with. Each cluster, for instance, is identified by a "cache path" that identifies it such that it can be fetched and rendered on any platform. In this example, the first cluster's cache path uniquely identifies it as cluster 1 of 5 clusters produced for harvest 4583. Deeper positions in a cluster tree are represented in the cache path by appending additional cluster identifiers. The XY coordinates in 2D space for this cluster are provided, as are the number of incident records in the cluster (5) and the terms that best describe that cluster (Miami, ship, and virus). Client implementations might display this information as a 2D display or as a folder structure showing nested clusters.

```

<ArrayOfClusters>
  <ClusterInfo>
    <CachePath>4583:5.1</CachePath>
    <X>0.4298539</X>
    <Y>0.9070176</Y>
    <DocCount>5</DocCount>
    <TopTerms>miami, ship, virus</TopTerms>
  </ClusterInfo>
  <ClusterInfo>
    <CachePath>4583:5.2</CachePath>
    <X>0.0496808961</X>
    <Y>0.6221393</Y>
    <DocCount>6</DocCount>
    <TopTerms>texas, van, hotel</TopTerms>
  </ClusterInfo>
  ...
</ArrayOfClusters>

```

Figure 1. Sample output from the SRS data clustering web services illustrates cluster-space description syntax.

To characterize the communication reduction benefits of the cache path and hierarchical clustering, consider a harvest of  $n$  documents. Transferring the full content of  $n$  documents or even the 2-space coordinates of the  $n$  documents to the client can become prohibitive as  $n$  grows. In a cluster tree, however, the amount of data transferred from server to client from top to bottom on one path is  $O(h)$ , where  $h$  is the height of the cluster tree, which is  $O(\lg n)$ .

### 3.5 Temporal theme identification

In addition to identifying clusters of related incidents, our users need to understand temporal changes in the attributes of these

incidents over time. The temporal granularity of interest varies with role; crime analysts might be most interested in temporal changes over periods of weeks or months, while field officers and scene commanders often want to spot trends within minutes to hours.

Using the approach developed for ThemeRiver [6], we have implemented a temporal theme detection service that describes the change in importance over time of attributes in source data. A `CreateTemporalVis` service can accept as input a cache path (which resolves on our server to a set of incident clusters and the top attributes for each) and bins the incident records it contains based on the time range occupied by the result set (shorter time windows are divided into smaller bins). The service returns to the client an XML document that describes, for each bin, a score reflecting the importance of each theme. Optionally, the client can request a theme analysis for a collection of specific terms of user interest or for a specific time window. Clients can use the latter parameter to produce zoomable temporal displays.

### 3.6 Interoperability with other services

The service-based analytic model allows information to be processed by a variety of third-party components before being rendered. This "mashup" approach to visual analytics removes many of the barriers to extensibility imposed by stovepiped, standalone applications. For instance, to produce geographic maps showing locations mentioned in text documents, SRS can call a web service running FactXtractor (described in [7]), a system for recognizing named entities, including places, in text documents. Once these placenames are identified, they can be passed to a geocoding service that determines the geographic coordinates for them, and finally back to the client for rendering.

## 4 ANALYTIC INTERFACES

The primary interface to SRS is via a web browser, described in Section 4.1. We also describe a mobile interface suitable for field officers in Section 4.2.

### 4.1 Web client overview

The web interface to SRS (Figure 2) includes seven visualization components with which users can explore incident collections. The client-side rendering of and interaction with each visualization is controlled by Javascript and dynamic HTML Document Object Model (DOM) manipulation. Interaction with SRS web services is performed through asynchronous Javascript and XML (AJAX) service calls. The available views include:

- A cluster space view that identifies groups of potentially related incidents based on shared attributes.
- A temporal theme view that shows trends in incident attributes over time.
- A faceted browser, which provides rapid drill down through result sets based on incident characteristics.
- A timeline that organizes incidents in chronological order.
- A map that shows the locations involved in each incident, and for each location, the number of incidents that occurred there.
- An entity-relation view, which allows analysts to explore associations between named entities in incident data.
- A list view that provides access to the original incident records.

The basic unit of analysis for each of these views ranges from the incident or document (in the cluster, timeline, and map views), to cross-cutting attributes of incidents or documents (in the temporal theme and faceted browser views), to specific entities within those incidents (in the entity-relation view). Support for this range of views gives users a choice of entry points to an analysis. Most of

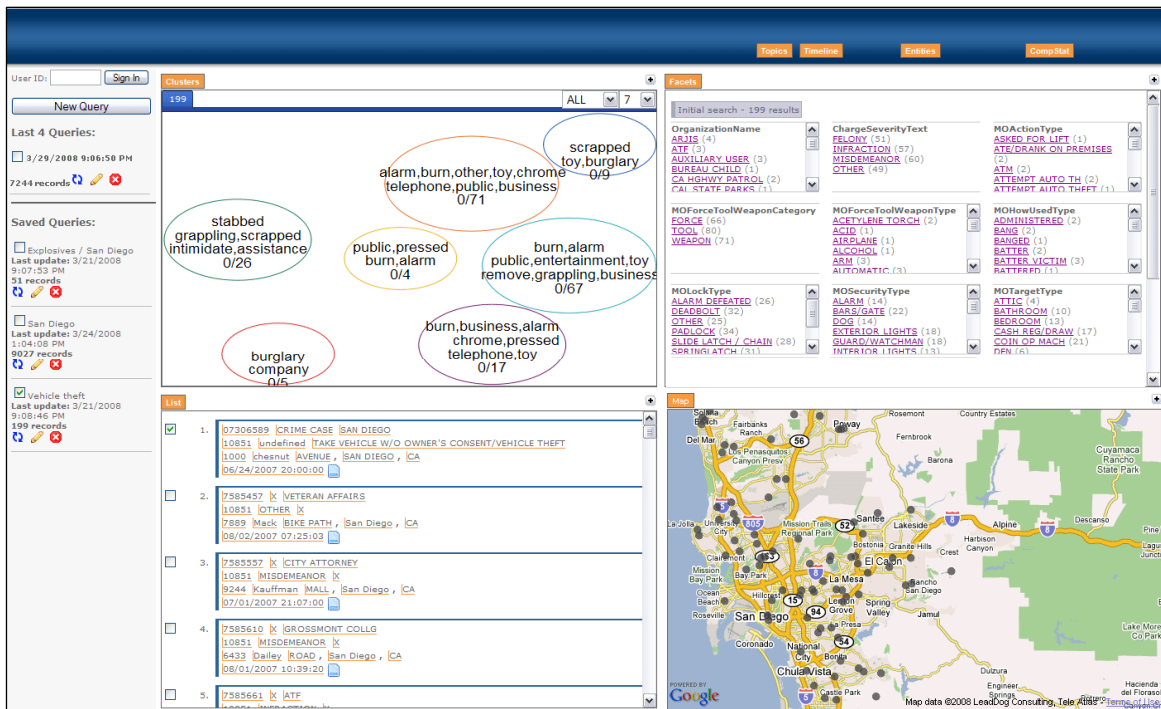


Figure 2. SRS web client showing, from top left, cluster, faceted, list, and map views of vehicle theft incidents.

these views are constructed via DOM manipulation, ensuring cross-browser compatibility. Some of the more complex views, including the cluster space, theme view, and entity-relation view, are rendered using Scalable Vector Graphics (SVG).

To achieve coordination across these views (enabling selections made in one visualization to be reflected in others), the SRS web client maintains a global selection model that broadcasts selection changes made in any view to every other view. This SRS client uses a focus and context model to facilitate incident exploration. *Selections* highlight the incidents that an analyst has marked as being of interest in any of the views. However, the full context of the result set is always shown, but de-emphasized. In Figure 2, no incidents have yet been selected – the full context is shown (note: in this and subsequent views, synthetic data is shown).

Along the left side of the display in Figure 2 is the current user's search history and saved queries. The query interface provides for dynamic creation and removal of search fields so that users can perform broad queries on only one or two fields or create highly complex queries with a combination of many fields. In the law enforcement context, search fields include approximately 45 incident attributes such as activity description, person description, vehicle description, and so on. Because many users have "profiles" over which they often search (e.g., a particular user might frequently look for new incidents near a location of interest), complex query conditions can be saved to a user's account allowing them to re-run them later. The user may execute previous queries from this list, modify a query, add new queries, or remove queries no longer needed. Every new query executed is added to a list of recent queries, automatically capturing an analyst's search history. This can be a powerful tool for testing various query scenarios to find the data needed.

Selecting any one of these queries loads its data into the main display; the checkboxes next to each query allow the user to create a union of multiple queries. In Figure 2, the currently active search is "Vehicle theft", which returned 199 incidents.

The main display shows one or more views onto the search results; here four views are open. The additional views are

available from a menu at the top of the screen and may be dragged on to the display; views may also be reordered or maximized.

A typical analytic session with the SRS web client begins by exploring the cluster and temporal theme views. Following the explore-enrich-exploit model, we have found that visualizations that display individual incident records (such as the timeline) or that require foreknowledge of individual features of interest (such as entities) are typically only used later in an analysis. Users first want to explore overviews of the general patterns in their data to become familiar with it; thus, the cluster display provides a summary of the primary groups of records in their search results, and a scan through the faceted browser shows the distribution of attributes across those results. Only after drilling into these visualizations to narrow the selection to a more manageable number of incidents do the analysts open up the "granular incident" views such as the timeline, map, or list.

#### 4.1.1 List view

Raw search results are shown in the list view (Figure 2, bottom left). The list includes a summary information block for each incident and a link to expand the entry to read the full incident record. A checkbox next to each incident selects it, causing the other views to highlight where this incident falls in their display. As selections are made in other visualizations, the list view automatically shuffles the selected incidents to the top of the list. Users requested this behavior to allow them to easily track two contiguous groups – incidents they are interested in, and those they are not – at the expense of not having a given incident maintain a static, memorable place in the list.

#### 4.1.2 Cluster view

At top left of Figure 2, the 199 incidents of vehicle theft have been clustered according to the process described in Section 3.4. Here, the system has grouped the incidents into seven clusters based on shared attributes. This view receives the cluster XML description sent by the `GetClusters` service and renders it in a 2D display. Each cluster is represented by an ellipse that is scaled to the number of incidents in the cluster; global relationships

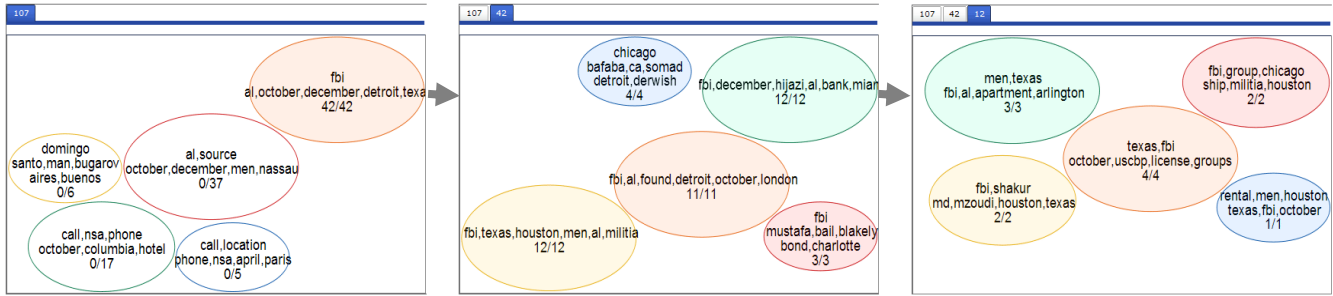


Figure 3. Drilling down through a cluster space (from left to right) identifies the incident groups that compose each cluster.

between clusters are preserved, such that nearby clusters are more similar than distant clusters. Each cluster is labeled with the incident attributes that best describe that group of incidents. Hovering the mouse over any of these terms provides additional information about the attribute for which it is a value; hovering over “assistance”, for instance, reveals that the modus operandi that these incidents share is that the perpetrator offered assistance to a stopped motorist. The numbers at the bottom of each cluster indicate the number of incidents in the cluster. The notation “0/26” illustrates how selections are kept in context. Of the 26 incidents in the cluster, none have yet been selected; if the user selects a group of incidents in any of the other views, the location of those incidents in cluster space will be indicated by changing the selection counts in the appropriate cluster labels.

Analysts can “steer” the clustering algorithm by specifying incident attributes that should carry more weight. The clustering can be steered by default categories of suspect description, vehicle description, or modus operandi, and analysts can also provide their own preferred attributes by which to steer. The result of steering by vehicle description, for instance, is that the cluster space is reorganized to show groups of incidents that are related by virtue of involving similar vehicles. Users can also manually control the number of clusters shown; selecting more clusters creates smaller groups of more closely related incidents, while selecting fewer aggregates incidents into more general categories.

After examining top-level clusters, the analyst then selects clusters to examine more closely. Clicking on a cluster executes a drill-down, where the incidents in the selected cluster are themselves clustered (according to the number and steering criteria the analyst has specified). Through iterative drill down, steering, and adjusting cluster numbers, analysts can pinpoint associations between incidents that would be difficult to identify by reading the full records. Incidents first clustered by modus operandi might allow the analyst to identify a set of crimes performed in a similar manner. Drilling into this cluster, the analyst might then ask the system to cluster that subset by suspect description, to see if the incidents have perpetrators in common.

Figure 3 illustrates the process of drilling down through a cluster space. In an initial cluster space containing 107 incidents, the analyst selects the cluster containing 42 incidents highlighted in red; the display updates to now show how just those 42 incidents cluster. The tab array at the top of the cluster visualization tracks the navigation path and allows the analyst to quickly traverse back up the cluster tree. Drilling down once more by selecting the cluster of 12 incidents at top right shows those 12 grouped into small clusters of very closely related incidents.

#### 4.1.3 Temporal theme view

Whereas cluster visualizations take an incident-driven approach to exploration, the SRS theme view allows trends across incidents to be identified. Figure 4 shows how an SRS web client may

render as an SVG graph the temporal theme information produced by the service described in Section 3.5, using the same data as in Figure 2. In this view, primary themes detected in the incident collection are shown in the legend at the left side of the chart; the user has filtered the list of themes to show just those that are tagged as names of organizations in the raw records, showing, in this case, the primary locations from which vehicles have been stolen. (Some of the less voluminous themes are not visible in this chart but may be explored by zooming in). The analyst notices that many of these organization names are those of schools, and that there was a large increase in the number of vehicle thefts at Henry High School (“henry” in the legend) in the days following January 11. Selecting a time window and a theme (multiple selections are possible, such that discontinuous regions can be selected) shows where the incidents that populate that theme fall in the other displays.

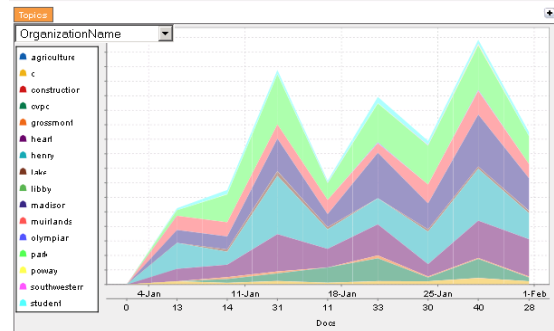


Figure 4. Temporal trends in the school locations where the fictional incidents of vehicle theft in Figure 2 occurred.

#### 4.1.4 Faceted browser

A faceted browser [8] is an interface for navigating categorical data efficiently. In SRS, structured data records are automatically organized into facets that show the distribution of values across each incident attribute. Clicking a facet value (top right in Figure 2), such as MOForceToolWeaponType = “Acetelyne Torch”, highlights, in the rest of the views, where the two vehicle thefts that involved this tool fall, and also causes the faceted browser to update its own display to show how the attributes of incidents that meet this condition are distributed. For instance, looking at the VehicleMake field after making this selection reveals that only Hondas were victim to this attack. As with the tabs in the cluster visualization, the faceted browser maintains a local breadcrumb trail that shows which facets are driving the current selection.

#### 4.1.5 Timeline and Map

To explore the chronological order of individual incidents, the timeline view (Figure 5, left) uses a third party web component<sup>§</sup>

<sup>§</sup> <http://simile.mit.edu/timeline>

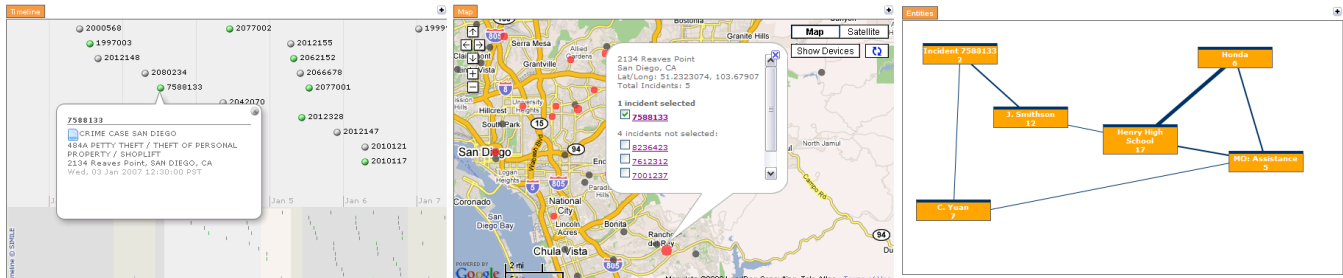


Figure 5. Left: A vehicle theft selected in the timeline. Center: The location of that theft, indicating that others have occurred at the same location. Right: An entity graph built around that location allows the analyst to discover additional suspects.

that allows multiple time scales to be displayed at once, and SRS automatically adjusts these scales to accommodate the granularity of the data. The timelines are compressed or attenuated to show less detail where few incidents occur and more detail when incidents are closely spaced in time. Individual incidents are shown with gray markers; selected incidents are shown in green.

In this view, the analyst has selected one of the incidents of vehicle theft at Henry High School identified through Figure 4. The map view (Figure 5, center) now shows where this incident occurred; it also reveals that there were four other incidents at this school, which are not included in the analyst’s current selection but may be worth investigation. On the map, incidents in the current selection are shown in red; the remaining context of unselected incidents is shown in gray. Keeping this context available is important for discovering potentially related information that falls outside of the parameters specified by the selections a user has made across views.

#### 4.1.6 Entity-relation view

Through exploration of the previous linked views, analysts often identify specific terms or entities of interest. In moving from the “explore” phase of their analysis to the “exploit” phase, they have winnowed a potentially large incident collection down to a smaller number to investigate more deeply. In this phase, a central aim is to discover relationships between entities, which may be people, terms, or other attributes of specific incidents such as locations or modus operandi.

Rather than automatically produce a large semantic graph showing all possible relationships in the data, we follow the approach of Jigsaw [9] in supporting manual exploration of graph relations. Our entity view begins as a blank canvas to which analysts add terms of interest as they discover them in other views. These terms may be names of people or organizations, or indeed any text string that occurs in an incident record.

In Figure 5 (right), the analyst has placed a node with the name of the school at the center; following that, nodes for incident attributes were placed, as was the name of one known perpetrator, J. Smithson. The analyst discovers that there is a prior incident in which this perpetrator committed an offense with C. Yuan, who has previously used the same MO (“feigning an offer of assistance”) as the thefts at Henry High School. The analyst is thus able to identify an additional subject of interest based on this analysis.

The web service that powers our entity view runs asynchronous queries against the incidents in the current result set; for each term the user enters, the service returns a node that indicates how many incidents contain that attribute. To produce edges between nodes, the service determines pairwise attribute co-occurrence scores. An edge is drawn between two attributes if they co-occur, and the thickness of the edge increases in normalized correspondence with the number of co-occurrences.

At this point, the analyst might propagate his or her selections to the cluster view in Figure 2 to determine if the incidents he or she has identified through the other views fall into the same cluster. If so, the analyst might explore that cluster to determine if additional vehicle thefts may be part of the same series.

#### 4.2 Mobile analytics

Desk-based investigators are not the only users who need access to the analytic functionality SRS provides. Field officers in our deployment organization need to quickly identify groups of related incidents using mobile devices they already carry, and on such devices it is impractical to scroll through long lists of text search results. The SRS data clustering services can be used to present a rapid visual picture of the structure of a result space.

Figure 6 shows a cluster visualization derived from the same data as Figure 3, but rendered on our Windows Mobile SRS client. Because screen space is limited on this client, it renders not 2D ellipses but a list of rectangular clusters containing the search results. Selecting a cluster drills down through a tabbed interface, as in the web client. In this fashion the officer can quickly see groups of incidents organized around common people, for instance, and is able to select the cluster of incidents that corresponds to the person he or she is interested in.

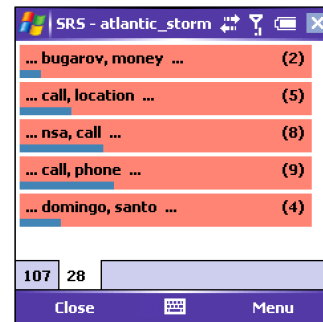


Figure 6. Incident clustering interface on mobile device.

SRS clustering services also enable a powerful method of performing complex queries from the field. Query-by-example (QBE) is a method for leveraging an existing cluster space to retrieve incident records. A typical use case for this approach is that an officer has just completed a field investigation (using a data input interface that is part of SRS but not shown here). Based on this investigation document, he or she now wants to know if the incident may be part of a crime series. The officer selects a “More like this” button on the investigation form, and the form is transmitted to the SRS server where it is decomposed into a term vector and projected into the 200-dimensional cluster space composed of existing incidents. Nearest neighbors in this space are identified by Euclidean distance, and the officer now knows whether the incident just described bears similarities to incidents that have occurred previously, even in other jurisdictions. QBE

helps build situational awareness by putting new information in the context of existing information.

## 5 SUPPORTING ANALYTICAL REASONING

When using visual environments such as SRS to explore data, there can be a gap between the interactive discovery process and the ability of the tool to record these discoveries so they can be shared, evaluated, and revised. Too often, the knowledge discovered through visual analysis is kept tacit in an analyst's mind or is recorded in a form disconnected from the discovery environment, such as a written document.

In SRS, analysts can graphically describe their knowledge construction workflows, linking hypothesis creation and testing to the visualizations that were used to derive these hypotheses. The SRS hypothesis construction workspace allows any feature of a view, or an entire view itself, to be saved as a "reasoning artifact".

Reasoning artifacts are essentially pieces of information that the analyst has tagged with a role; these roles are defined through a customizable taxonomy of knowledge structures modified from [10]. Currently, this role taxonomy includes such elements as *evidence*, *assumption*, *spatial or temporal pattern*, *causal relationship*, and *hypothesis*. These roles were refined through discussions with our analysts and represent the common high-level tags they apply to information.

At any point during an analysis, a user can open a reasoning whiteboard on the SRS web site and begin to create artifacts that link features in SRS views to reasoning roles. Features such as incident clusters can be dragged out of the view and onto the whiteboard to capture them as a reasoning artifact, where they are converted to a small sticky note (Figure 7). Annotation artifacts can also be created to record information such as assumptions. Any reasoning artifact can be used as an edge to connect other artifacts; for instance, an analyst might place a blank artifact describing asserting that the suspect in two crimes is the same person, and use this argument to link together two artifacts containing separate clusters of incidents. Examining the map showing the location of incidents near the school, the analyst might articulate a hypothesis that the perpetrator is a student at the school. This hypothesis is recorded as a new artifact and linked to the emerging reasoning graph.

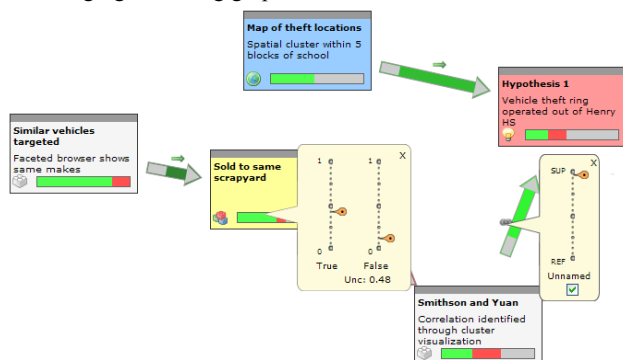


Figure 7. A reasoning graph, with confidence and support estimation sliders open, enables analysts to record hypotheses and supporting evidence as they explore SRS visualizations.

To begin evaluating the strength of their hypotheses, analysts can describe their confidence and uncertainty in evidence and the degree to which each node in the graph supports or refutes their hypotheses. A bar graph on each reasoning artifacts represents its current belief state, and clicking on the graph allows the analyst to use sliders to update this state. The reasoning whiteboard automatically propagates confidence and support scores across the graph to produce a visual depiction of how the analyst's confidence scores in each artifact combine to produce likelihood

estimates for hypotheses, using the Dempster-Shafer belief network approach described in [11].

## 6 RELATED WORK

The advantages of distributed, web-based analytic systems are reflected in the growing number of applications that follow this model. GeoBoost [12] offers mapping, charting and collaboration capabilities via web and mobile clients and a server-side syntax for including data in client views, although SRS provides an open library of analytic services as well as coordinated visualization capabilities. COPLINK [13], a query and analysis tool widely used in law enforcement, provides some graph-based and spatial visualization capability through a web interface, but it does not provide automated clustering, reasoning support, or coordinated visual exploration of search results.

Oculus TRIST [14] and Sandbox [15] provide powerful mechanisms for querying, organizing, and reasoning with complex data. The Sandbox is a model for integrating a variety of visual analysis interfaces into an exploratory knowledge capture environment; SRS follows this approach, but uses a distributed web infrastructure to enable analysts to explore data collections from any web browser. We also emphasize a lightweight approach to visualization, creating visual interfaces suitable for deployment over distributed networks.

FacetMap [16] offers a visual interface to faceted browsing of large data collections that addresses some of the scalability concerns in navigating with facets; it provides a solution that adaptively changes the facet depth displayed according to the faceted browsers across display sizes.

There is also a long-standing thread of research in rule-based visualization, of which our service-based approach is a variant. Rogowitz and Treinish [17] propose rules that guide the production of visual displays based on perceptual principles and faithful representation of underlying data. Early approaches to service-based visualization transferred complete visualization descriptions (e.g., [18]) to the client, which offloaded computationally intensive processing but did not result in client-independent visualization.

The support SRS offers for the creation of reasoning graphs that describe an analyst's findings is similar to that found in visual argumentation tools such as that in ClaiMaker [19] and Compendium [20], although SRS allows these argument structures to be created within the same application that powers the data-driven investigations.

## 7 DISCUSSION AND FUTURE WORK

SRS currently offers a suite of client-server visualization components that can be composed into interactive web applications for online, lightweight visual analytics. During preliminary evaluation of SRS as part of its law enforcement deployment, we have identified several areas where SRS provides benefits heretofore unavailable to investigators, analysts, and field personnel, and still other areas where further work is needed.

Providing multiple visualization components from which users can select when exploring incident data allows these users to tailor their choice of views to the task at hand. They can change views as they move from exploration to exploitation phases of analysis, and the range of components available through SRS supports examination of both broad patterns (e.g., through cluster views) and specific relationships (e.g., through entity views).

Feedback from our users has guided design decisions; for instance, initially the default selection state for all incidents was "on", and users would winnow down to a set for further exploration by deselecting incidents, locations, or attributes that they were not interested in. However, users preferred to work in the opposite mode. Starting from a context overview where all

incidents are deselected, a few incidents, attributes, or clusters are “turned on” to determine where they fall in the other views. By user request, we are also implementing a global “breadcrumb trail” that tracks the selection history of all views. These breadcrumbs allow users to track the process by which they arrived at a set of incidents. They can also move forward and back through their interaction history, recreating the state of the environment at any point during their exploration.

We are also currently evaluating an alternative data clustering algorithm, QROCK [21], which is designed for structured categorical information such as our law enforcement records and may provide more meaningful clustering of incidents. An outstanding challenge, however, remains the seamless integration of both structured and unstructured data into a single exploratory environment. Incident data from some providers comes in the form of narrative text, but linking incident attributes across different formats requires that new approaches to information extraction be integrated into the SRS platform.

Finally, our entity view does not currently support more sophisticated named-entity recognition (NER) and automated semantic graph generation, but that is a limitation of the backend service that powers it rather than of the client. Our approach is limited in that the graph can only show nodes for terms the user has expressly asked for. A powerful feature of the SRS approach, though, is that an NER service could replace the current query-driven approach, and as long as the XML response format remains the same, the client visualization need not change. This ability to make instream changes to alternate analytic services is a core benefit of a service-driven toolkit.

## 8 CONCLUSIONS

To support information analysis in organizations where the analytic function is distributed across many users, new scalable visual analytic architectures are needed that enable deployment of lightweight applications. The Scalable Reasoning System is a service-oriented analytic toolkit that decouples computationally intensive data processing from client interaction, allowing visual analytic applications to be deployed through web browsers, mobile devices, and other platforms that support pervasive access across an enterprise. SRS clients can support multiple views that enable large, web-accessible data sets such as search results to be explored through coordinated interaction. Initial application of this system is in the law enforcement domain, and we are currently expanding SRS deployments to other regional law enforcement and homeland security organizations across the U.S.

Our long-term goal is to support a community-wide, open library of analytic services. We anticipate that the service-based analytic model will enable tool developers to rapidly deploy analysis applications that make use of “gold standard” algorithms without having to re-implement those algorithms. Ultimately, an interoperable service library will bring to visual analytics the same mashup capability that has made other web applications, such as geographic mapping, so successful.

## ACKNOWLEDGEMENTS

This work was supported by the National Visualization and Analytics Center (NVAC), a U.S. Department of Homeland Security Program operated by the Pacific Northwest National Laboratory.

## REFERENCES

- [1] Viegas, F., M. Wattenberg, F. van Ham, J. Kriss, and M. McKeon, "ManyEyes: A site for visualization at internet scale," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, pp. 1121-1128, 2007.
- [2] Chittaro, L., "Visualizing information on mobile devices," *Computer*, vol. 39, pp. 40-45, 2006.
- [3] Pattath, A., R. May, T. Collins, and D. Ebert, "Real-time scalable visual analysis on mobile devices," in *Conference on Multimedia on Mobile Devices, 20th Annual IS&T/SPIE Symposium on Electronic Imaging*. San Jose, CA, 2008.
- [4] Patterson, E., E. Roth, and D. Woods, "Predicting vulnerabilities in computer-supported inferential analysis under data overload," *Cognition, Technology, and Work*, vol. 3, pp. 224-237, 2001.
- [5] Hetzler, B., V. Crow, D. Payne, and A. Turner, "Turning the bucket of text into a pipe," presented at IEEE Symposium on Information Visualization (INFOVIS 05), Minneapolis, 2005.
- [6] Havre, S., E. Hetzler, P. Whitney, and L. Nowell, "ThemeRiver: Visualizing thematic changes in large document collections," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, 2002.
- [7] Pan, C. and P. Mitra, "FemaRepViz: Automatic extraction and geo-temporal visualization of FEMA National Situation Updates," presented at 2007 IEEE Symposium on Visual Analytics Science and Technology (VAST), Sacramento, CA, 2007.
- [8] Hearst, M., "Clustering versus faceted categories for information exploration," *Communications of the ACM*, vol. 49, pp. 59-61, 2006.
- [9] Stasko, J., C. Goerg, Z. Liu, and K. Singhal, "Jigsaw: Supporting investigative analysis through interactive visualization," presented at 2007 IEEE Symposium on Visual Analytics Science and Technology (VAST), Sacramento, CA, 2007.
- [10] Johnston, R., "Developing a taxonomy of intelligence analysis variables," *Studies in Intelligence*, vol. 47, 2003.
- [11] Sanfilippo, A., B. Baddeley, C. Posse, and P. Whitney, "A layered Dempster-Shafer approach to scenario construction and analysis," presented at 2007 IEEE Conference on Intelligence and Security Informatics, New Brunswick, NJ, 2007.
- [12] Eick, S., M. Eick, J. Fugitt, B. Horst, M. Khailo, and R. Lankenau, "Thin Client Visualization," presented at 2007 IEEE Symposium on Visual Analytics Science and Technology (VAST), Sacramento CA, 2007.
- [13] Chen, H., D. Zeng, H. Atabakhsh, W. Wyzga, and J. Schroeder, "COPLINK: Managing law enforcement data and knowledge," *Communications of the ACM*, vol. 46, pp. 28-34, 2003.
- [14] Jonker, D., W. Wright, D. Schroh, P. Proulx, and B. Cort, "Information triage with TRIST," in *2005 International Conference on Intelligence Analysis*, 2005.
- [15] Wright, W., D. Schroh, P. Proulx, A. Skaburskis, and B. Cort, "The sandbox for analysis: Concepts and evaluation," presented at Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Montreal, 2006.
- [16] Smith, G., M. Czerwinski, B. Meyers, D. Robbins, G. Robertson, and D. Tan, "FacetMap: A scalable search and browse visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, pp. 797-804, 2006.
- [17] Rogowitz, B. and L. Treinish, "An architecture for rule-based visualization," in *IEEE Conference on Visualization*. San Jose, CA, 1993.
- [18] Trapp, J. and H.-G. Pagendarm, "A prototype for a WWW-based visualization service," in *Proceedings of the Eighth Eurographics Workshop on Visualization in Scientific Computing*, 1997.
- [19] Mancini, C. and S. J. B. Shum, "Modelling discourse in contested domains: A semiotic and cognitive framework," *International Journal Of Human-Computer Studies*, vol. 64, pp. 1154-1171, 2006.
- [20] Buckingham Shum, S., V. Uren, G. Li, J. Domingue, and E. Motta, "Visualizing internetworked argumentation," in *Visualizing Argumentation*, Kirschner, P., S. Buckingham Shum, and C. Carr, Eds. London: Springer-Verlag, 2003, pp. 185-204.
- [21] Dutta, M., A. Mahanta, and A. Pujari, "QROCK: A quick version of the ROCK algorithm for clustering of categorical data," *Pattern Recognition Letters*, vol. 26, pp. 2364-2373, 2005.