EventShop: Recognizing Situations in Web Data Streams

Siripen Pongpaichet
University of California, Irvine
spongpai@ics.uci.edu

Vivek K. Singh
Massachusetts Institute of Technology
singhv@mit.edu

Mingyan Gao
Google Inc.
gaomingyan@gmail.com

Ramesh Jain
University of California, Irvine
jain@ics.uci.edu

ABSTRACT

Web Observatories must address fundamental societal challenges using enormous volumes of data being created due to the significant progress in technology. The proliferation of heterogeneous data streams generated by social media, sensor networks, internet of things, and digitalization of transactions in all aspect of humans' life presents an opportunity to establish a new era of networks called Social Life Networks (SLN). The main goal of SLN is to connect *People* to *Resources* effectively, efficiently, and promptly in given *Situations*. Towards this goal, we present a computing framework, called EventShop, to recognize evolving situations from massive web streams in real-time. These web streams can be fundamentally considered as spatio-temporal-thematic streams and can be combined using a set of generic spatio-temporal analysis operators to recognize evolving situations. Based on the detected situations, the relevant information and alerts can be provided to both individuals and organizations. Several examples from the real world problems have been developed to test the efficacy of EventShop framework.

Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: Online Information Services - *Data sharing, Web-based services*; H.2.4 [**Database Management**]: System - *Multimedia databases*, *Query processing*; H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems

General Terms

Management; Design; Human Factors

Keywords

EventShop; Events Processing Framework; Situation Recognition; Web Data Streams; Web Observatory; Social Life Networks

1. INTRODUCTION

During the first-generation of the Web (Web 1.0) in 1990s, the focus was primarily on building the Web, and making it accessible by connecting *people* to online *documents*. In the the second-generation (Web 2.0) in 2000s, the growth of social networking sites, wikis, communication tools and folksonomies brought a new experience to our society, by connecting *people* to *people*. In addition, the emergence of the mobile Internet and mobile devices was a significant platform driving the adoption and growth of the Web. Effectively, the Web became a universal medium for data, information, and knowledge exchange. In the third-generation Web (Web 3.0), the innovation shifted toward upgrading the back-end Web infrastructure level making the Web more connected, more exposed, and more intelligent. The Web is transformed from a network of separately siloed applications and content repositories to a more seamless and interoperable as a whole. The Web now can be used to establish a new network called "Social Life Networks (SLN)" [11] by connecting *people* to real-world (life) *resources* for decision making at both individual and societal levels.

The decisions making can range from daily life problems such as traffic, to emergency situations such as hurricane migration. With the enormous reach of mobile phones, used by more than 5 billion people, SLN can be used to connect people with life resources even in the remote areas of underdeveloped countries. Real world phenomena are now being observed by multiple media streams which are available in real-time over the Web and increasingly the majority of these has space and time semantics. Let's take hurricane migration case as an example: there are massive volumes of related heterogeneous data stream available on the Web such as hurricane status (NOAA.gov), weather forecast (weather.com), population demographics (census.gov), rescue shelters (redcross.org), and traffic directions (maps.google.com). Despite rapidly increase of data available on the Web, comprehensive development tools and computational frameworks for effectively combining and processing these available heterogeneous streams are lacking. Solving particular problems such as giving a direction to the nearest shelter to the people who are currently in the unsafe areas are very difficult to achieve and time consuming by just manually browsing on the Web. To provide appropriate resource to individuals and organization efficiently and promptly, the most important problem is to recognize the situations happening around them.

Recognizing situations in the right place at the right moment to take appropriate actions can save lives and resources in various aspects of human life. For instance, in healthcare domain, approximately 250,000 people die prematurely each year from asthma attacks [5], and almost all of these deaths are avoidable. Asthma patients should have been notified to avoid the places that can cause their asthma attacks when they are in critical situation. Some of the early examples studied such as flood alerts in Thailand [8], Flu monitoring [18], and new business store location [19] clearly demonstrate the importance of situation recognition and the beneficial of SLN.

WWW 2013 Companion, May 13-17, 2013, Rio de Janeiro, Brazil.
ACM 978-1-4503-2035-1/13/05.

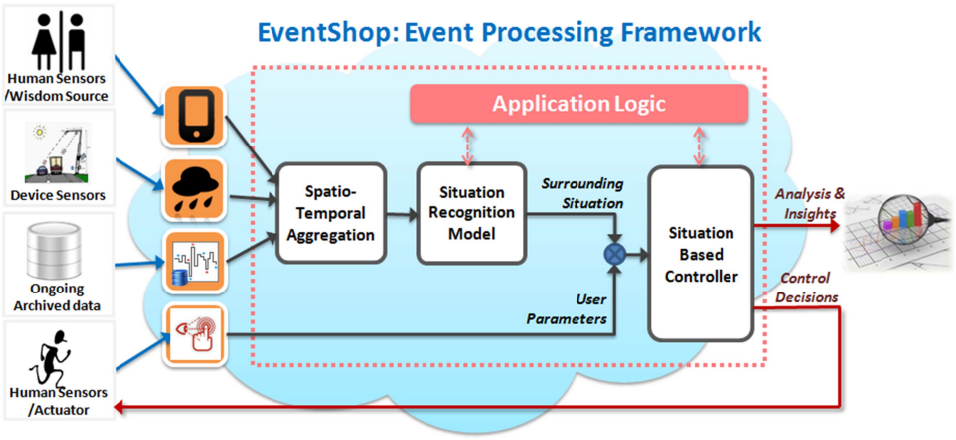


Figure 1: "EventShop" Event Processing Framework towards "Social Life Network"

Building toward SLN, a novel computational framework called "EventShop" has been designed and developed to integrate and process on heterogeneous Web data streams. The overall framework is shown in Figure 1. The data streams over the web (e.g. tweets, weather.gov feeds) are translated into a unified format and made amenable for computing in the next step. Based on application logics, multiple spatio-temporal analysis operators are formed to generate different situation recognition models. The uniform data streams are continuously fed into the models to detect and recognize real-time situations. Then, the detected situations (e.g. 'flu outbreak' in New England) can be combined with user parameters (e.g. 'high temperature', and location). Again, based on application logic, the situation based controller is constructed to send out personalized action alerts (e.g. 'Report to CDC center on 4th street') to each individual. In addition, the analytic reports are also available to a central analyst who can then take large-scale (state, nation, corporate, or world-wide) decisions.

The challenges for this framework come from two parts. First, the data ingestion to efficiently extract data from the Web and make them available for later computation is not-trivial. The data on the Web are not only generated in different media format (e.g., KML, JSON, image, table, and sensor signal), but the properties of them are also very different (e.g., measuring weather, and traffic). While semantic web technologies and APIs are trying to make data interoperable, there is still a lack of a generic data model which can be used to integrate heterogeneous data coming from spatio-temporally distributed web streams. Second part is the stream processing engine to bridge the semantic gap between high level concept of situations and low level data streams.

The solution to our main problems lies in realizing the fundamentally different nature of these Web data streams related to real-world situations. Majority of the data now can be defined as a particularized property located at some region of space-time. For all situations, space can be used to capture their proximity, while time can be used to analyze their procession. Therefore, space and time become the most important key aspects to integration and processing of different data streams. By using a simple unified representation based on ***S**pace-**T**ime-**T**heme* (STT), EventShop indexes and organizes all data into a common representation called STT Observation. A basic assumption in this approach is that spatio-temporal situations are determined by evaluating a large number of data streams that represent different attributes measured by either physical sensors or observed by human-sensors. To define and detect complex situation over vast number of data streams, the processing engine should be programmed using assorted spatio-temporal operators. There is no doubt how important situation recognition in big data is. The more data we can get from the Web, the more accurate situation recognition will be. We believe that space, time and theme should be the center attributes for sharing data over the Web. This simple but very significant requirement can bring the Web to the next evolution but it has been always overlooked.

In summary, the key contributions of the work reported here are as follows. First, we propose and design the computing framework to recognize real world situations and generate appropriate information and action from heterogeneous spatio-temporal data streams available on the Web. We describe a generic approach to integrate and operate data streams based on their space, time and theme (where, when, and what). Second, we define a useful set of operators commonly used in spatio-temporal analysis to bridge the semantic gap between high level situation concept by the human and low level data on the Web. Finally, we develop a system prototype which allows users from multiple domains to easily build diverse applications in experimental environment. We ensure that the EventShop computational framework can take lead on the spatio-temporal streams analysis and contribute to a new vision of the Web where data are connected. We anticipate multiple situation-aware applications to benefit from having these Web data available in term of space and time.

2. RELATED WORK

Our framework has really emerged from a marriage between the several erstwhile areas of event-based computing, Geographical Information Systems (GIS), web service choreography, semantic web, and mashup tools.

Multiple recent efforts have looked at analyzing social and sensor data for understanding the relevant situations. Data Stream Management System (DSMS) and Complex Event Processing (CEP) systems, e.g. Rapide [12], provide sophisticated support for data stream analysis but typically focus on data in the cyber space only. Their main applications' focus is different from our focus. The Geographical Information System (GIS) provides handful operators on spatial analysis, but are limited on temporal analysis on streaming data.

There also has been a growing interest in web service choreography [1] and semantic web services [13]. However, most of these efforts focus on well-structured (typically ontological)

data. We on the other hand make the unified representation extremely simple (STT), which allows many web streams to integrate easily. Semantic web technologies have been trying to make content on the Web meaningful. Their analytics tools have primarily focused on theme or entity relationships, but slightly support analysis on spatial and temporal relationships which are often the critical components for recognizing situation. To extract data from the web pages, several techniques have been practically used including screen scraping (i.e., HTML Parser) and deepweb [3]. The former aims to extract the content from the surface web pages, while the latter try to access the content on invisible or dynamic web pages.

Mashup and domain specific mashup tools (e.g. MashArt [6], http://pipes.yahoo.com) also try to integrate data from multiple web contents. However, most of web mashups remain very shallow (only visual integration) and do not provide sophisticated (e.g. geo-temporal) analysis capabilities. Efforts like Yahoo Query Language, Google Tables, Yahoo Pipes [23], and MashArt provide easy, modular, computational tools for users to integrate and analyze web data. This resonates deeply with our EventShop system. Following similar lead, our ultimate goal is to provide a computational framework for sophisticated spatio-temporal analysis of heterogeneous web streams to undertake situation based control.

3. OVERALL FRAMEWORK

There are two main components in EventShop framework which are *Data Ingestor* and *Stream Processing Engine*. A workflow of moving from heterogeneous raw data streams to actionable situations is shown in Figure 2. In the Data Ingestor component, original raw spatio-temporal data from the Web are translated into unified STT (Space-Time-Theme) format along with their numeric values using an appropriate *Data Wrapper*. Based on users' defined spatio-temporal resolutions, the system aggregates each STT stream to form an E-mage stream which we will describe in more detail later in this Section. E-mage was first introduced in [19]. These E-mage streams are then transferred to the *Stream Processing Engine* component for processing. Based on situation recognition model determined by the domain expert, appropriate operators are applied on the E-mage streams to detect situation. The final step is a segmentation operation that uses domain knowledge to assign appropriate class to each pixel on the E-mage. This classification results in a segmentation of an E-mage into areas characterized by the situation there. Once we know the situation, appropriate actions can be taken. Next, we will describe the challenges in building this framework.

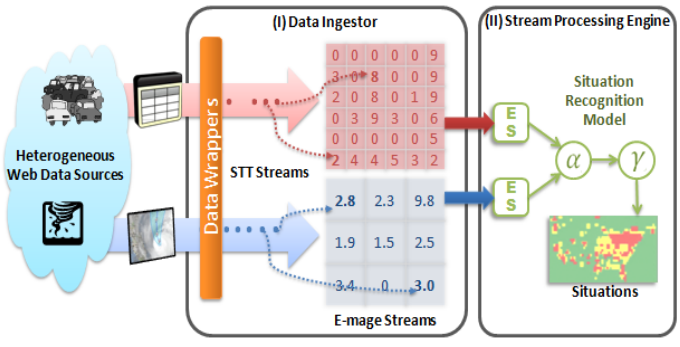


Figure 2: From Heterogeneous Data Sources to Situation Recognition

3.1 Heterogeneous Data Sources

By understanding the fundamental nature of the data streams observing real-world events, we found that each data observation is always associated with a time and geo-spatio metadata [Perry 2008]. This commonality between different streams motivates our modeling approach. There are several approaches for modeling spatial, temporal and thematic (STT) data; for example, using Semantic Web data models [17, 15], Spatiotemporal Database [20], and GIS Spatio-Temporal Modeling [21, 14]. Given the geo-spatial continuity, we believe that a spatial grid structure is naturally suitable for representing various geo-spatial data, where each cell of the grid stores value of certain measure taken from the corresponding geo-location. We adopt the grid structure, and call it E-mage (an event data based analog of image). We believe that this generic data model can be used to integrate heterogeneous data coming from spatio-temporally distributed web streams.

For handling data streams, special attentions need to be paid to the semantics of aggregated data. Since the data streams are unbounded, combining such a data to find simple aggregated value such as summation and average is unclear. This problem is normally resolved by introducing windows which transform an unlimited sequence of data streams into overlapped or non-overlapped windows of data. In this work, we use *tumbling* [4] window that splits data streams into non-overlapped contiguous windows.

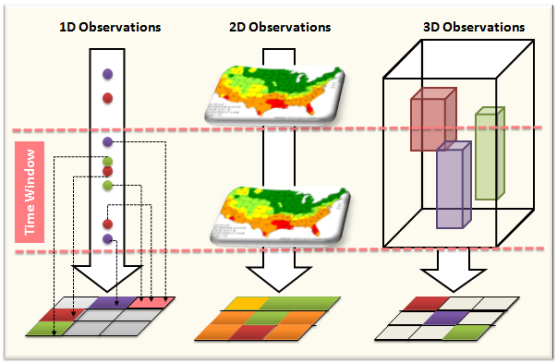


Figure 3: From Heterogeneous STT Observations to E-mage

Apart from various media formats of the data, we can classify spatio-temporal input streams into three categories based on their behavior as shown in Figure 3. First, the simplest and the most common type is 1D (or point) STT Observation. Each data observation measures a property in the real-world at particular point in time and geo-location. Data point examples are tweets and sensor signals. For each the time window, STT Observations are collected to generate one E-mage. Based on the E-mage's resolution, and the spatial coordinate of STT Observation, each data point is spatially mapped to a cell in the E-mage. The value at the cell is normally the `sum` of values of all the STT Observations mapped to the cell. Depending on the applications, different aggregates can be applied such as *count, min, max,* and *average.*

Second category is 2D STT Observation. The values of the observations are already aggregated over time and space. This type of data is generally found on the geo-images (i.e., pollen count from http://pollen.com/images/usamap.gif), maps, and most of the KML data format. In this case, the value at a cell in an E-mage is computed from the original or normalized values at the pixels of the original geo-image that are projected to this cell. Computation of projected area depends on the geo coordinate

projection system. The last data type is 3D STT Observation. Unlike 1D STT, each data captures the real-world property for a period of time at a region (not a point). Thus, each cell in an E-mage often consists of multiple values from many STT Observations. The interpolation and convolution techniques to constructing the cell from the set of data observations are required [16].

3.1.1 Data Unification Model

Each E-mage represents some measure taken over a certain geographical area in a time window, and the measure can be evaluated from various heterogeneous data sources, including social sources, such as Twitter, Facebook and Flickr, as well as other geo-graphical sources. For example, we collect tweets about a given topic, e.g. "Obama", from Twitter. Then for each time window, an E-mage can be constructed, each cell of which represents aggregated number of tweets regarding this topic coming from the corresponding geo-location. An example E-mage representing interest level amongst users across mainland US in terms of number of tweets containing the term "iphone" on 11th Jun 2009 is shown in Figure 4.

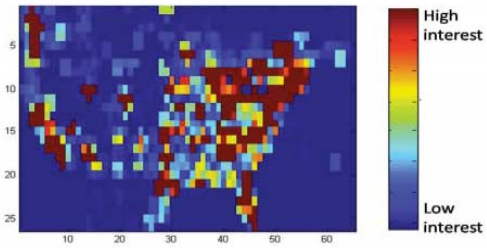


Figure 4: An Example E-mage

Although the use of a single number in each cell to represent a measure seems a very simple model, combining multiple such E-mages that characterize different measures of an event could potentially generate very insightful results and discover interesting phenomena hidden behind. This approach can capture the semantics and the notion of spatial neighborhood very elegantly, and geography-driven-joins [10] between data reduce to simple overlaying of grids. Humans are quite used to seeing satellite images and GIS data on similar interfaces, so they can understand such data much better than any text-centric representation. E-mage allows intuitive visualization and hence aids situation awareness for a human user. In addition, computation on grid-like structure can efficiently speed scale up in several ways, including parallel computing with GPU, and spatial index (i.e., R-tree [9], quad-tree [7] and KD-tree [2]).

3.1.2 Data Representations

All observed spatio-temporal data stream coming into the EventShop framework are converted to, and represented in a common **S**pace, **T**ime, and **T**heme (STT) format. A STT Observation is represented as:

```
STT = <latitude,longitude,timeStamp,theme,value>
```

A flow of STTs becomes a STT Stream:

$$\text{STT Stream} = \{STT_0, \ldots, STT_i, \ldots\}$$

Space is captures in *latitude* and *longitude*. The time is *timestamp*. The *theme* is a description of what this data observed. The *value* is a single numeric value. Values in STT Observations collected in a particular time window over STT Stream can be aggregated to form a two-dimensional data grid. The data grid together with related STT information is called *E-mage*, represented as:

```
E-mage = < SWCoord, NECoord, latUnit, longUnit,
          TimeStamp Theme, 2D Grid >
```

SWCoord and *NECoord* are the southwest and northeast spatial coordinate to which the value at the bottom-left and the top-right cell in the 2D grid corresponds. *latUnit* and *longUnit* specify the actual spatial granularity distance, such as 40 miles or 0.1 unit of the width and height of each cell in the grid. The *timeStamp* of an E-mage is the end time of the time window over which the STTs are collected, which is normally equal to the timeStamp of the last STT in the time window. A flow of E-mages forms an *E-mage Stream*:

$$\text{E-mage Stream} = \{\text{E-mage}_0, \ldots, \text{E-mage}_i, \ldots\}$$

Building block of the 2D grid in an E-mage is a single cell. The cell together with STT information is called *stel* (spatio-temporal element),

```
stel = <SWCoord, NECoord, latUnit, longUnit,
        timeStamp, theme, value>
```

A stel is a special E-mage, where the size is 1 by 1. Some operators take normal E-mage stream as an input and generate an output of this special E-mage stream, called *stel stream*:

$$\text{stel Stream} = \{\text{stel}_0, \ldots, \text{stel}_i, \ldots\}$$

3.2 Operators for Situation Recognitions

3.2.1 Common Operators

We analyzed situation recognition models across multiple domains and defined the initial set of operators, which are generic enough to capture most of the common requirements. We expect to keep enriching this set as this framework gets applied to more applications. Note that these operations are used to find out interesting E-mage through describing their characteristics, rather than to manipulate them directly. Therefore, these operations are more declarative than the underlying operations in image processing.

3.2.1.1 Selection (σ)

One common task in our work is to select E-mages or part of an E-mage from E-mage stream that satisfy certain predicates. To meet this requirement, we design the selection operation σ to support the following predicates, a *value* range, *spatial* bounding box, and *time* interval.

3.2.1.2 Segmentation (γ)

This operator segments each E-mage in a single E-mage Stream. The current methods are *K-means* and *Thresholding*. The segmented E-mage can be either separated into multiple E-mages or created a single E-mage with cell values corresponding to the assigned segment.

3.2.1.3 Aggregation (α)

Aggregation operator takes two or more E-mages stream as an input and combines them to generate a new E-mage stream as an output. E-mages are aggregated per cell. For example, for each *(i, j)* pair, the sum aggregation adds *cell(i, j)* of E-mage from each input stream, and stores the sum value at *cell(i, j)* of the new E-mage. We assume that size of E-mage from all input streams are the same. In addition, this operator allows one or more operands to be scalar or 2D pattern, in which case, every E-mage in E-mage streams can be combined with the scalar or the pattern.

Aggregates supported are *max, min, sum, avg, sub, mul, div, and, or, not, xor, convolution.* With some exceptions, "*sub"* and "*div"* function can only be applied between two E-mages, and "*not"* function is only allowed on one E-mage.

3.2.1.4 Spatial Characterization (∅)

Spatial characterization operator takes a single E-mage stream and computes a representative stel to characterize this E-mage, based on a characterization function. The output is a stel stream. Each stel stores the spatial coordinate where the measure is taken, and the associated value. Characterization measures can be selected from *max, min, sum, avg, epicenter,* and *coverage*. For functions as *max, min,* and *epicenter*, the stel is the one where the value is reached. For instance, after using max, the stel whose value is the largest among all others in the E-mage is returned. However, for other functions such as *sum, avg, coverage* and *periodicity*, the computation produces only a value without any specific location point.

3.2.1.5 Spatial Pattern Matching (φ)

Spatial pattern matching operator takes an E-mage stream and a two-dimensional pattern as inputs, and tries to match the pattern in every E-mage. The output of this operator is a stel stream. The output stels record the location of highest match, and the corresponding similarity value.

3.2.1.6 Temporal Characterization (τ)

Unlike previous operators, temporal characterization operation is designed to characterize stel stream instead of E-mage stream. It takes a window of stels from a stel stream, and computes its characteristics like *displacement, velocity, acceleration, periodicity,* and *growthrate.* The output of this operator is again a stel stream. We know that stel stream is the result of spatio-temporal characterization or pattern matching on E-mage streams. Sometimes we may want to study how these characterization values or pattern similarities vary over time. For instance, given an E-mage stream, we first find out epicenter of each E-mage in the E-mage stream, and then we may want to know how the center point moves from time t_0 to time t_k, where k + 1 is the size of time window. To use the pattern, users need to specify the time window size k, over which the stels are characterized.

3.2.1.7 Temporal Pattern Matching (ω)

Temporal pattern matching operator compares the values of a window of stels in stel stream to certain 1D pattern, and see how similar the two trends is. Similarly, to use the operator, users need to specify the time window size. Output of this operator is a stel stream, where each stel stores the center position of the sub window of stels where the pattern matches with the highest similarity value, and the similarity value. The value 1.0 means that the sequence of stel values and the given pattern match with each other completely. Currently, we support *linear, exponential,* and *periodic* patterns matching. For generating a pattern, the users need to specify the sampling rate and the duration. For example, the sampling rate could be 1 value per 5 seconds, and the whole pattern duration is 30 seconds. For *linear* pattern, parameters include slope and Y-intercept. *Exponential* pattern need the base value and the scale factor and *Periodic* patterns use frequency, amplitude, and phase delay.

Note that in the current set of operators, the spatio-temporal operators (e.g. temporal pattern matching on velocity of epicenter of a hurricane) are handled by applying temporal operators on the outputs of spatial operators. The summary of supported operators is shown in Figure 5.

Operation	Input	Output
Selection	E-mage Stream	E-mage Stream
Segmentation	E-mage Stream	E-mage Stream
Aggregation	K* E-mage Stream	E-mage Stream
Spatial Characterization	E-mage Stream	Stel Steam
Spatial Pattern Matching	E-mage Stream	Stel Steam
Temporal Characterization	Stel Steam	Stel Steam
Temporal Pattern Matching	Stel Steam	Stel Steam

Figure 5: Summary of Operators

3.3 Towards Web Observatory

The main goal of Web Observatory [22] is to create a distributed archive of data on the web. As a result, state of the Web in terms of topologies, resources, links and activity, in the past, present, as well in the future can be examined, observed, and predicted. However, most of data on the Web do not only represent information in the cyber space, but also capture the real state of the physical world. This offers other great opportunities to various domains including real world situation recognition. Although, data held by the Web Observatory are so heterogeneous in terms of media formats and properties, data used in situation recognition can often be described along three dimensions: spatial, temporal and thematic.

In EventShop, we provide several data wrappers to continuously retrieve data on the Web (in various formats) and translate them into STT streams and E-mages stream respectively. Enormous spatio-temporal data on the Web are collected and made available by Web Observatory. Instead of creating specific *wrapper* for each individual web page, it would be more useful to develop STT data *adapter* to bridge Web Observatory data source and EventShop. Similar to in traditional database like MySQL, data adapter serves as a bridge between a data set and a data source for retrieving and storing data. Querying data over space, time and theme attributes will be fundamental requirements for this adapter. We have started to see this functionality implemented on many popular web APIs such as Twitter, Facebook, Flickr, and New York Time News.

We already proposed a generic approach for integrating spatial, temporal and thematic (STT) data from heterogeneous data streams and converting them into unified E-mage streams for further processing. We also defined a set of common spatio-temporal operators that can be applied on the stream of E-mages to detect complex situations. Next, we will describe our prototype system and demonstrate that EventShop is a suitable platform for analyzing the vast quantity of data captured by the Web Observatory.

4. EVENTSHOP ARCHITECTURE

The EventShop system takes spatio-temporal data streams as inputs, and allows users to register situation models (queries) into the system by using the E-mage operators as presented in Section 3. Result streams of queries are finally outputted and shown to the end users. A generic system as EventShop would allow users with different study backgrounds to experiment with their spatio-temporal data streams, and potentially saves repeated efforts from building similar hard-coded systems. In this section, the system requirements and architecture are described.

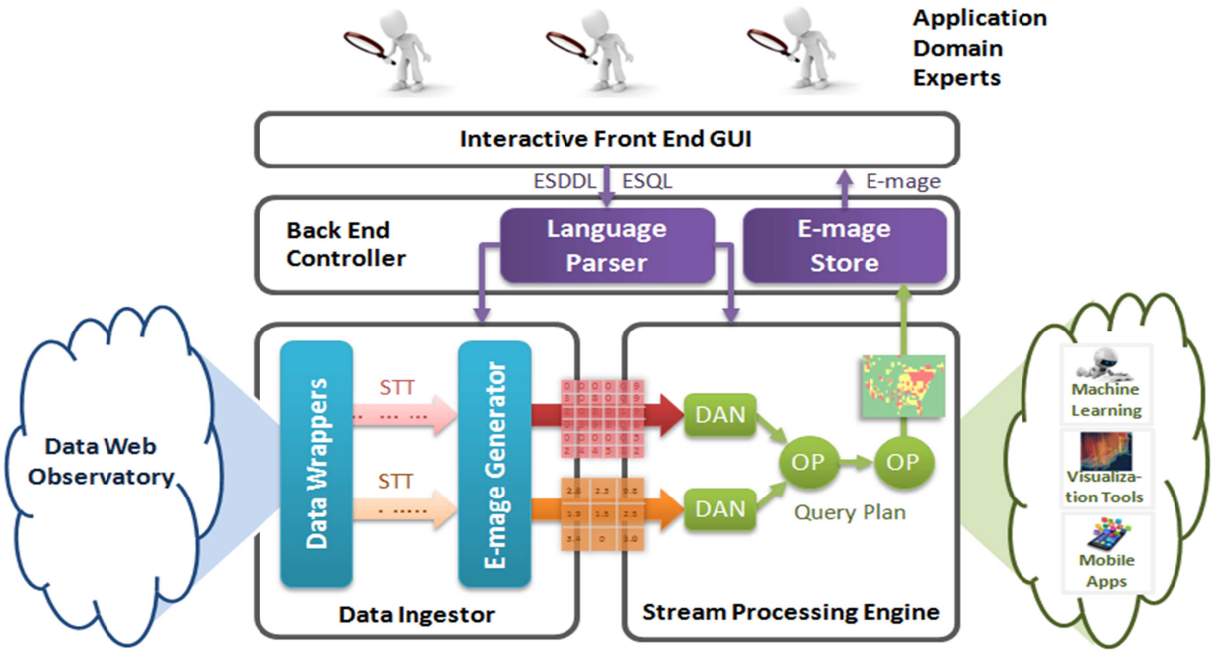


Figure 6: EventShop System Architecture

4.1 System Requirements and Design

There are number of system requirements needed to be satisfied by the design of EventShop.

- The processing engine has to be able to handle data flowing into the system as continuous streams. Unlike traditional database or batch processing engine, EventShop needs to deal with standing query over continuous streaming data.
- The data integration needs to handle highly heterogamous data streams on the Web. As shown in the previous section, given the inherent spatial semantics of these streams, we have proposed the unified data structure, E-mage, to model and combine heterogeneous streams of aggregated spatio-temporal data.
- The spatio-temporal data streams provided by different data sources are not only heterogeneous in terms of media, but also often available at different spatio-temporal bounding boxes and granularities. Approaches that can handle such differences among streams need to be designed.
- EventShop is not a system for specific domain application, but it is a computational framework for the Web in general. Thus, we need to design the system in generic fashion so that it is domain independent and hence can be employed in many types of applications. As a result, a set of useful generic operators for spatio-temporal analysis are defined and built-in the EventShop system.
- Lastly, the system need to provide a graphical user interface (GUI) that is user-friendly and encouraging for end users to easily experiment with their data.

4.2 System Architecture

The EventShop system architecture is presented in Figure 6.

EventShop includes both a front end GUI as well as a back end engine. Inspired by the user-friendly GUI of PhotoShop which allows users to experiment on their photos, the *front end GUI* of EventShop allows end users to register new data stream sources and formulate queries by combining a rich set of built-in operators. User interactions on the GUI widgets trigger events that are directed to the corresponding event handlers by the JavaScript engine. These event handlers get parameters values from the events and generate JavaScript objects, including data source objects and query objects, to temporarily store the information. When submit event is triggered, these front end objects are serialized into certain JSON format, according to the specification of the EventShop Data Definition Language (ESDLL) and EventShop Query Language (ESQL). These JSON message are sent to the backend Servlet server through Ajax call for backend processing. The result E-mage can be fetch from the *E-mage Store* to display on the front-end GUI.

In the back end, data sources and queries objects represented in ESDDL and ESQL are received at *Backend Controller* which passes them to *Language Parser*. The parser parses these objects according to the language, and creates backend data source or query objects that are used by *Data Ingestor* and *Stream Processing Engine* respectively. These backend objects are also saved in registered data source and registered query main memory storage. According to the data source object specification, appropriate *Data Wrapper* is selected to ingest and convert raw heterogeneous spatio-temporal web data into uniformed STT Observations. These data stream transmitted to *E-mage Generator* to create E-mage streams. Meantime, directed by the registered queries, stream processing engine pulls E-mage streams from e-mage generator and processes the E-mage streams in each of the instantiated query operators. The outputs E-mage streams are stored back to *E-mage Store* at the back end controller.

There output E-mage, such as classified E-mages, can be directly sent to end users for situation recognition and action taking. However, users (i.e., application domain experts) can apply other analysis operators and tools, such as machine learning toolkits, to further process, and analyze the data. E-mage streams can also be visualized using variety of visualization tools to visually discover patterns, trends, and anomalies between situations besides analyzing spatial-temporal correlation. Since the output E-mage

streams can be accessible over the Web, the usages of them are not limited to our front end GUI. For example, the asthma mobile phone application can combine the global environments situation with users' personal data on the phone (i.e. activities level, location, and asthma level), and send out personalized message to individuals such as enjoy outdoors, avoid outdoors, take medication, visit doctor, and so on.

4.3 Data Ingestor

After a data source is registered, parsed and inserted into the data source storage, back end controller creates new instances of *STTObservationIterator* as well as *EmageIterator* (as described later in this section) for this data source and adds them to the data ingestor. Data ingestor then connects to the data source and takes the raw spatio-temporal-thematic data stream as input, and relies on these iterators to convert the raw data stream into an E-mage stream.

4.3.1 Data Source Specification

A registered data source needs to include the following information to enable data ingestion process:

a) *Theme*: the main topic discussed in a data source such as flood, hurricane, asthma, flu, population, pollution, and etc.
b) *URL*: the API access link of a data source. For example, Twitter opens its Stream and Search API which allows users to query tweet streams. Sensor data, such as traffic sensors provided by PeMS (Caltrans Performance Measurement Systems), is also publicly available on the Web.
c) *Data Type*: Raw data stream are available to access in different formats and granularities. Several data inter-change formats are already supported such as, JSON, CSV, KML as well as geo-image (PNG, JPG, BMP, GIF). Beyond data media formats, input data streams are categorized into three types which are STT stream, geo-image stream, and array stream. For Twitter and many sensors, single data point, such as a tweet and signal, is the unit generated and appended to data stream. Some data sources aggregate STT data and offer them in geo-image format, for instance, pollen count data (http://pollen.com/ images/usamap.gif). Other data sources provide a set of data collected in a time window in array format. This type of data gets updated at each time window and forms an array stream. A data source needs to be declared as one of the types above.
d) *Type Specific Parameters*: Different data types require different specific parameters.
e) *Frame Parameters*: To aggregate STT stream into E-mage stream, the parameters for specifying the size, resolution and generation frequency of an E-mage are needed. This set of frame parameters consist of the size of time window (e.g. 10 seconds, 1 hour, 1 day), the synchronization time point (e.g. creating an E-mage at every 10th second, or at 6AM everyday), unit of latitude (0.1 latitude = 40miles), unit of longitude, spatial boundary including southwest and northeast coordinates. The frame parameters used to guide the framing of raw STT stream to create E-mage stream is called Initial Frame Parameters. The final Frame Parameters will be introduced later as required by processing query.

4.3.2 System Design of Data Ingestor

As for now, all integrations are done in main memory to guarantee low latency and rely on iterators life cycle to continuously generate E-mage. A data ingestor is comprised of two types of iterators, STTObservationIterator and EmageIterator.

a) STT Observation Iterator

A STTObservationIterator interface is a iterator that each raw data point into STT format, and outputs one STT Observation at each `next()` function call. Specific wrappers to handle particular data sources can be implemented from this interface. For example, for each incoming tweet in the Twitter stream of a specific topic `T`, (e.g., hurricane), along with the meta-data associated with the tweet, the Twitter wrapper finds out `tweet time` and `tweet location`, and generates one STT `<tweet location, tweet time, T, 1>`. Value in the result STT is customized by the wrapper. It could be the count, e.g. 1, of the tweet, or the temperature value. We are also opening APIs that allow users to implement their own wrappers to ingest more data streams.

b) E-mage Iterator

At each time window specified in the initial frame parameters, an EmageIterator generates one E-mage. This E-mage remains in its internal buffer until query processor pulls the E-mage by calling `next()` function. Based on the data source type, E-mages could be generated from STT stream, geo image stream and array stream.

4.4 Stream Processing Engine

After registered data sources, the situation recognition model can be formed by applying different spatio-temporal operators on any of data sources, and registered into the system. This model is represented as a rooted query plan tree (logical operator tree) to show an ordered set of steps for accessing or processing on E-mages. Nodes of a query plan tree are either Data Access Node (DAN), or Operation Node (OpNode). Leaf nodes of a query plan tree are always DAN, and internal nodes are OpNode. Root node of a query plan tree represents the last operator in the plan applied to an E-mage. Edges between nodes represent the E-mage stream flowing from a lower level node to a higher level node. [an example shown in section 5]. The model or query plan is different from traditional one-time query issued to database because it a standing query. A standing query needs to be registered and instantiated in the system before related data streams flow into the system and get processed. For each operator of each registered query, as specified by its parameters, back end controller creates an operator instance that performs the actual computation.

Next, back end controller connects these operator instances as defined in the logical operator tree of the query and forms a runtime operator tree in the stream query processor. Then, as required in the query, controller pulls E-mage streams from the EmageIterators (in data ingestor) of the appropriate data sources, and feeds the streams to the runtime operator tree. Each operator instance processes the E-mage streams pulled from upstream operators and buffers the results in its internal buffer. The E-mage stream output from the last operator instance is the output of the entire query.

A query registered by end users needs to specify two parts of information, a *Final Frame Parameters* and a logical operator tree. Queries may need to access the same data source but combine them with other data sources at different spatial bounding box, resolution or time window. Therefore, every query specifies a set of final frame parameters, which should be followed by all input E-mages streams. E-mages pulled from data ingestors need to be first mapped to E-mages following the final frame parameters.

The system has a *Resolution Mapper (RM)* component, which takes an E-mage stream of a data source, the corresponding initial frame parameters, and final frame parameters as requested by a query as input, and generates a new E-mage stream following the final frame parameters. In the RM component, we allow users to select the strategy to perform frame mapping. If E-mages at coarser frame parameters are mapped to finer frame parameters, the set of strategies includes `interpolation`, `repeat`, and `split`. If the mapping is performed on the other way, users can choose strategy like `sum`, `max`, `min`, `avg`, and `majority`, for conversion.

4.5 Web GUI

The Web front end GUI of EventShop is shown in Figure 7. It consists of five main components as follow:

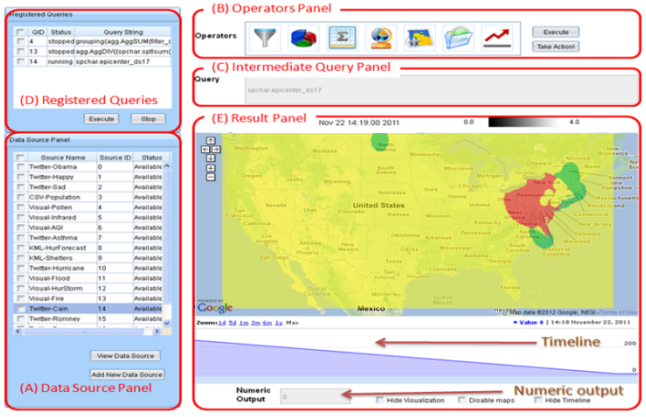


Figure 7: EventShop Web GUI

a) *Data Source Panel*: displays a list of data sources that registered with the system. Users can click on each data source to view its description and its recent E-mage created by the system. Users can also register a new data source into the system.

b) *Operators Panel*: allows users to form a query by applying different operators on any of the data sources, as well as allow them to configure action control to send out alerts. The built-in spatio-temporal operators are categorized into seven groups as mentioned in Section 3.

c) *Intermediate Query Panel*: shows a textual representation of the intermediate query currently being composed by the user.

d) *Registered Queries Panel*: Displays a list of standing queries that registered with the system. It allows users to start and stop the query processes.

e) *Result Panel*: displays the recent output of the running query, E-mage and stel, which can be presented on a map, timeline, numeric value output, or a combination of them.

5. USE CASES

For the past couple years, several applications have been designed and developed using the EventShop framework to recognize real world situations across multiple domains; for example, Hurricane detection and migration, Identifying demand hot-spots of business products, Flu outbreak, Wildfires detection, Flood migration, and Allergy risk and recommendation. We select two applications and demonstrate here.

5.1 Thailand Flood

This application aims to suggest the safe location to people who are trapped in Thailand flood on October – November 2011.

To achieve the goal, we first classify the flooding areas into three groups based on flooding condition and shelter sufficiency. Then for those people who tweeted about flood from the dangerous zones, we sent tweets to them and directed them to the nearest shelter in the safe areas. The data sources include the map of flood affected are across Thailand (www.thaiflood.com/floodmap) and the shelter location map (www.shelters.thaiflood.com). Both data sources are continuously collected every few hours in the KML format, an international standard format used to display geographic data on the Google map. Moreover, we collect tweets send for central of Thailand with keywords both in Thai and English including "#น้ำท่วม", "#ThaiFlood", "#Flood", and "#ThaiFloodEng". The logical query plan of the grouping query and parameters of each operator are presented in Figure 8.

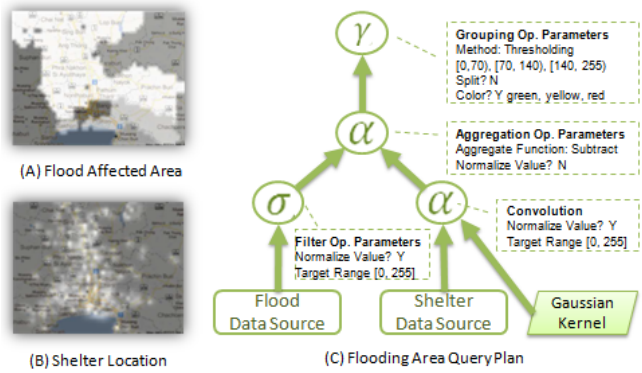


Figure 8: Sample E-mages and Flooding Area Query Plan

The final result of this grouping query is a stream of E-mages. Finally, we use the E-mage result along with the tweet data source to send out personal alert tweets. Grouping-query result as well as a snapshot of our twitter account sending tweets is shown in Figure 9. Some of our tweets are being re-tweeted by tweet receivers. Our Twitter account is @SocLifeNetworks.

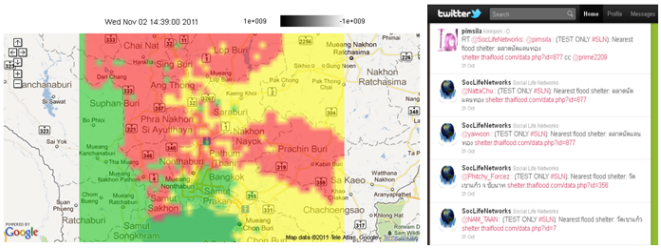


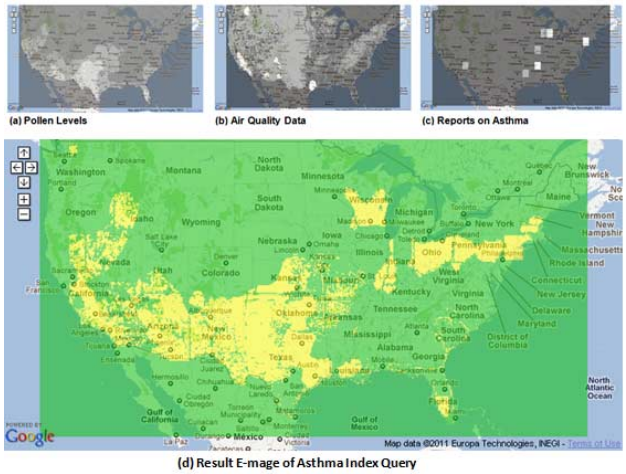
Figure 9: E-mage Result and Alert Tweets Sent from EventShop

5.2 Asthma Relief Application

In this experiment, we combine data from three data sources related to asthma study, and then segment the aggregated data over entire US into three danger zones based on the values in aggregated E-mages.

The severeness of an environment to an asthma patient is related to the pollen count and air quality in that area. From crowd sourcing aspect, if many people from a specific area discuss about asthma, it usually suggests that the situation in that area is not very friendly to asthma patients. The keywords selected for the Twitter source include "asthma" and "allergy". Figure 10 (a, b, c) shows sample E-mages from the three data sources.

The final frame parameters specified for this query are <1 day, 0, US, 0.1 lat x 0.1 long>. E-mages coming from the three data sources with distinct initial frame parameters are mapped to E-mages following the same final frame parameter by Resolution Mapper. The final E-mages (shown in Figure 10(d)) will be generated every day at 12AM with the spatial resolution as 260 x 590.



(d) Result E-mage of Asthma Index Query

Figure 10: Data Source E-mage and Result E-mage of Asthma Application

6. CONCLUSION AND FUTURE RESEARCH

Web contains enormous volumes of evolving spatio-temporal thematic data. The goal of a Web Observatory is to provide a framework to ingest, combine, process, and utilize this data for various societal applications. Based on the importance of space, time and theme, we proposed that these attribute should be accessible across the web. We envision that EventShop and similar other computational frameworks will have an active role in the evolution of Web 3.0. We designed and developed EventShop platform for processing heterogeneous Web streams.

An important aspect of the real system with multiple data streams which is not addressed in this paper is the quality or trustworthiness of data sources. The confident values of the data source, as well as new distributed E-mage will be implemented in the next version to solve this problem. In another aspect, we have lightly used the benefit of "theme". Taxonomy or ontology on the theme should be explored to help us in data source selection over massive set of web streams. In addition, we are in the process of making this framework available as an open source around the mid-year of 2013.

REFERENCES

[1] Arkin, A. et al. 2002. Web Service Choreography Interface (WSCI) 1.0.

[2] Bentley, J.L. 1975. Multidimensional binary search trees used for associative searching. *Commun. ACM*. 18, (1975), 509–517.

[3] Bergman, M.K. 2001. White Paper: The Deep Web. Surfacing Hidden Value. *The Journal of Electronic Publishing*. 7, (2001), online.

[4] Carney, D. et al. 2002. Monitoring streams: a new class of data management applications. *Proceedings of the 28th international conference on Very Large Data Bases* (2002), 215–226.

[5] Cruz, A.A. 2007. *Global surveillance, prevention and control of chronic respiratory diseases: a comprehensive approach*. Who.

[6] Daniel, F. et al. 2009. Hosted Universal Integration on the Web: The mashArt Platform. *ICSOC/ServiceWave* (2009), 647–648.

[7] Finkel, R.A. and Bentley, J.L. 1974. Quad Trees: A Data Structure for Retrieval on Composite Keys. Acta *Inf.* 4, (1974), 1–9.

[8] Gao, M. et al. 2012. Eventshop: from heterogeneous web streams to personalized situation detection and control. *WebSci* (2012), 105–108.

[9] Guttman, A. 1984. R-Trees: A Dynamic Index Structure for Spatial Searching. *SIGMOD Conference* (1984), 47–57.

[10] Hjaltason, G.R. and Samet, H. 1998. Incremental Distance Join Algorithms for Spatial Databases. *SIGMOD Conference* (1998), 237–248.

[11] Jain, R. and Sonnen, D. 2011. Social Life Networks. *IT Professional*. 13, (2011), 8–11.

[12] Luckham, D.C. 2012. *Event processing for business : organizing the real-time enterprise*. Hoboken, N.J. John Wiley & Sons.

[13] McIlraith, S.A. et al. 2001. Semantic Web Services. *IEEE Intelligent Systems*. 16, (2001), 46–53.

[14] Parent, C. et al. 1999. Spatio-temporal conceptual models: data structures+ space+ time. *Proceedings of the 7th ACM international symposium on Advances in geographic information systems* (1999), 26–33.

[15] Perry, M.S. 2008. *A framework to support spatial, temporal and thematic analytics over semantic web data*. Wright State University.

[16] Peuquet, D.J. 2001. Making space for time: Issues in space-time data representation. *GeoInformatica*. 5, (2001), 11–32.

[17] Sheth, A.P. and Perry, M. 2008. Traveling the Semantic Web through Space, Time, and Theme. *IEEE Internet Computing*. 12, (2008), 81–86.

[18] Singh, V.K. et al. 2010. From microblogs to social images: event analytics for situation assessment. *Proceedings of the international conference on Multimedia information retrieval* (Philadelphia, Pennsylvania, USA, 2010), 433–436.

[19] Singh, V.K. et al. 2010. Social pixels: genesis and evaluation. *ACM Multimedia* (2010), 481–490.

[20] Wang, X. et al. 2000. Spatiotemporal data modelling and management: a survey. *Technology of Object-Oriented Languages and Systems, 2000. TOOLS-Asia 2000. Proceedings. 36th International Conference on* (2000), 202–211.

[21] Yuan, M. 1996. Temporal GIS and spatio-temporal modeling. *Proceedings of Third International Conference Workshop on Integrating GIS and Environment Modeling, Santa Fe, NM* (1996).

[22] Web Observatory Wiki. http://wow.west.webobservatory.org

[23] Yahoo pipes. http://pipes.yahoo.com/pipes/