DEVELOPMENT OF AN EVEN-DRIVEN AND SCALABLE OIL SPILL MONITORING AND MANAGEMENT SYSTEM

H. Assilzadeh [a, *], Z. Zhong[b], A. S. Kassab [a], Y. Gao [a] J. K. Levy [c],

[a] Department of Geomatics Engineering, University of Calgary, Alberta Canada - (hassilza, ygao)@ucalgary.ca
[b] National University of Defense Technology, ChangSha, HuNan, China - nongnudt@gmail.com
[c] L. Douglas Wilder School of Government and Public Affairs Virginia Commonwealth University

The Role of Geomatics in Decision-Making: Providing Real Value to Canada's Economy and Society

KEY WORDS: KEY WORDS: Event-Driven System, Monitoring and Management, Oil Spill, Laser Fluorosensors

ABSTRACT:

This paper describes the development of an event-driven and scalable oil spill monitoring and management system. The system provides functions for managing oil spill information. The system differs from the traditional web-based request/response interaction model. Specifically, advantages of an event-driven system are its abilities to automatically match events with the subscriptions from the clients and push the event information to related clients in real-time when events occur. The event-driven system is also scalable: this means that it can easily be extended to include new functions or to integrate new information services. The system's middleware spatially processes the rules and algorithms for events and supports real-time data transaction from heterogeneous sources. GIS-based analytical information processing modules generate the oil spill monitoring and management output products. Laser fluorosensors are incorporated in order to enhance oil spill detection and surveillance. Other system components include a central repository system, database and a web application interface.

1. INTRODUCTION

Oil spill management requires oil spill information including spill location, and the size and extent of the spill. It is also important to understand the oil type and spill trajectory information. Remote sensing is an important oil spill monitoring tool which can help to detect oil spills and to extract oil spill information before spills cause widespread damage. Laser Fluorosensors, such as the Scanning Laser Environmental Airborne Fluorosensor (SLEAF) sensor operated by Environment Canada, are among the most appropriate remote sensing sensors for oil spill surveillance for their ability to detect oil on all backgrounds including ice, snow and soil. There are several advantages to laser fluorosensors: they have a high spatial resolution; a high temporal resolution, and the capability to classify different types of oil. These sensors are also capable of day and night operations and constitute the only sensor which can positively detect oil on shorelines (Jha, 2008).

It is very important to quickly detect oil spills and to distribute real-time information to users (including first responders, residents, fishers and tourists). A traditional information system generally adopts a request/response communication model, which is based on a point-to-point, synchronous and pulling mode interaction between the consumer clients and the information providers/services. On the other hand, scalable system architecture allows for the oil spill system to be easily extended by new function modules or integrated with other information sources to meet new oil spill response requirements. An event-driven communication infrastructure adopts a publish/subscribe interaction mode: users subscribe to events and the system automatically notifies users and "pushes" event information to users in real-time when events are generated by publishers. The event-based system architecture can integrate a wide range of observational technologies and monitoring instrumentation and can solve important oil spill monitoring and management challenges. A significant advantage of the event-based architecture using a publish/subscribe model (compared to current request/response model) is its capability to support real-time data transaction from heterogeneous sensors to end users (Eugster et al., 2003). Such a system can support many-to-many data communications and services since the intelligent middleware system handles all published notifications and "pushes" all data to interested consumers in an asynchronous and in real-time fashion (Keramitsoglou et al., 2004). The decoupling of interacting clients from the communication responsibility is advantageous for two reasons. First, it provides a high level of system adaptability. Second, it makes the integration of autonomous and heterogeneous components in distributed systems easy to scale and evolve. The use of a publish/subscribe mode along with a distributed GIS and enterprise database technology can improve oil spill monitoring and management systems by integrating many stationary and mobile devices or sensors for use by distributed users with heterogeneous interests. This paper describes the development of an event-driven and scalable oil spill monitoring system that allows for real-time information dissemination and transactions. The system will support efficient oil spill detection and response for improved decision making. A prototype system has been developed and the functions of different system components will be described in this paper. The system performance will also be discussed along with areas for future improvement.

2. EVENT-DRIVEN SYSTEM ARCHITECTURE

The publish/subscribe interaction model (Figure 1) is used in event-based computing (Muhl et al., 2006). The key concept of this interaction paradigm is the introduction of a middleware component which facilitates the interaction between the distributed clients. The middleware component is responsible for conveying information messages from the producer clients (publishers) which generate and publish event information to consumer clients (subscribers), who are interested in receiving

* Corresponding author.

the events' information messages. In this form of interaction, the clients are loosely coupled from each other. In other words, they interact without direct knowledge of each other. The producers publish event messages (notifications) which might be of interest to the clients. The consumers receive only events that they have already registered for (subscriptions). The middleware, called the event-notification service, handles the published events, matches them with the registered subscriptions, and pushes the events to matched subscriber clients asynchronously and in a timely manner. This interaction procedure enables heterogeneous, autonomous, and dynamic clients or sensor devices to be integrated in the system and leads to better scalability and communication efficiency.

Figure 1: Publish/Subscribe system components

There are four different subscription models (i.e., filtering models): channel-based, topic-based, type-based, and content-based models (Muhl et al., 2006). We discuss the topic-based and content-based models since they are the most widely used in practice. The first subscription model to be adopted in event-based systems is the topic-based, or subject-based notification (Oki et al., 1994, TIBCO, 1999). With a topic-based mechanism, the notification service predefines a set of subjects or topics by which notifications and subscriptions are classified. Producers are able to publish notifications to any of the predefined topics by annotating each of their notifications with a name string or an ID that refers to a certain topic. On the other hand, consumers receive published notifications relating only to topics for which they have subscribed. Industrial solutions, such as Vitria M3O (Holloway, 2008), TIBCO Enterprise Messaging Bus (TIBCO, 2000), and USENET News system (Harrison, 1995), have adopted the topic-based mechanism. An XML model for describing a schema for topics has been investigated as part of the web notification service standard (Graham et al., 2004). The content-based subscription model is the most generic notification selection mechanism (Muhl, 2001). This mechanism allows subscribers to express their interest not only in the topic, but also in the actual content information from the notifications. In fact, the topic-based model is considered a special case of the content-based model in which the topic name string of notifications is evaluated against register subscriptions (Cao, 2006). Using the content-based model, a subscription encapsulates a single or conjunctive predicates (i.e., boolean-valued expressions) that constrain the notifications of interest. Those predicates are evaluated over the content information of the notifications and matching information is delivered to customers. A simple predicate usually contains an attribute name, a basic comparison operator (e.g., =, >, <, ≥, ≤, LIKE), and a value in the same data type of the attribute name. More complex subscriptions can be formed by combining more than one predicate using logical operators (e.g., AND, OR).

For oil spill monitoring applications relevant information includes information about the geospatial locations of the spills or weather-related information. The clients may wish to express their interests using either the topic or content based subscription model. In a topic based subscription model the clients will receive all oil spill information even if the information is not requested (or useful). On the other hand, in a content based subscription model, the users can specify their interest about specific oil spill information. In a topic based event-driven systems, the event (oil accident) data may be in the form of text data, or even a file. The user's subscription can encapsulate a single or batch information about the oil spill event that constrains the relevant information about the event (Figure 2).

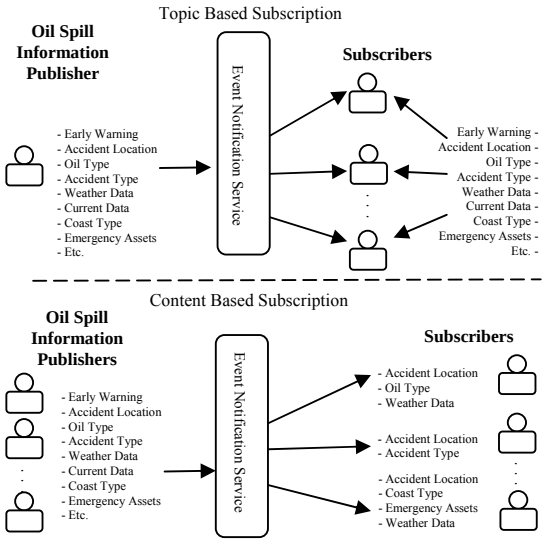

Figure 2: Topic based and content based event subscriptions

3. GEOSPATIAL-BASED PUBLISH/SUBSCRIBE

Since the communication middleware in an event-driven architecture (topic-based or content-based) cannot process the geospatial events, this research will extend the event-driven architecture functionalities to support the geospatial events-based publish/subscribe model and apply it to oil spill monitoring and management.

To accommodate geospatial events that are associated with the 2D geographic domain, basic comparison operators (e.g., =, >, <, ≥, ≤, LIKE) and logical operators (e.g., AND, OR) filters may not be sufficient to express spatial constraints. In the context of geospatial events, spatial filters are needed to allow consumers to specify their interests in a certain geospatial context. A geospatial notification includes two data components: (a) a spatial component and (b) an attribute component. The spatial component contains the geometric shape and location of the event and is used to describe the spatial semantics of an event. The spatial component is hence considered the predominant data component of a geospatial notification and can be represented by one of the following basic spatial features: a point, a line, or a polygon. The attribute component is a collection of attributes used to assign descriptive information about the circumstances or the properties of an event. Each attribute has a distinct name, a data type, and a data value. Data types include numeric, string, boolean, date/time, and bytes. They can be utilized to attach broad kinds of descriptive information to a geospatial notification. Here, the binary data type supports digital files, including images, documents, media files and other types, to be serialized and encapsulated in the content data of geospatial notifications. For instance, an airborne remote sensing sensor captures images of an oil accident scene, where in each image a geospatial notification is initialized. The captured dataset is serialized to a binary format and assigned to a binary type attribute. Finally, the geospatial notification is published to

notify other interested clients about the current status of the oil spill event. Extending the scope of publish/subscribe events to include geospatial semantics is a promising approach for numerous web GIS applications. In specific, oil spill emergency response systems require the immediate flow of geospatial information to the correct people. However, associating geospatial semantics with events in publish/subscribe systems invokes new challenges to the underlying communication infrastructure. Thus, the proposed model, namely a geospatial-based publish/subscribe system attempts to provide a suitable interaction framework for transacting geospatial events between distributed clients. This is described in more detail in the following sections.

3.1 Geospatial Event Clients and the Interaction Flow

The development of geospatial-based publish/subscribe model is conducted in three phases which include designing the geospatial data notification model by which producer clients publish geospatial events/notifications; designing the geospatial subscription data model which is used by consumer clients to subscribe their interests in geospatial notifications; and developing the matching method to evaluate published geospatial notifications with registered geospatial subscriptions and finding the matched subscribers. Publish/Subscribe data models comprise the geospatial notification, data model and the geospatial subscription data model. Clients are supposed to utilize these data models in their publications or subscriptions respectively. In notification data model geospatial notifications comprises a set of name/value pairs. Each pair specifies a single attribute of the associated geospatial event. Formally, a geospatial notification n_g is formed by a set of nonempty attributes (a_1, a_2 … a_n), where each a_i is a name/value pair (n_i, v_i). Each name n_i is assumed unique in the attributes set and has a single data type associated with it. The value v_i should be assigned according to the data type of the name n_i. The data types supported for constructing the name/value pairs are similar to the SQL data types, briefly: string, integer, float, boolean, date/time, and byte-array. In addition to that, geometry data types, including: GeometryPoint, GeometryPolyline, GeometryPolygon, GeometryMPoint, GeometryMPolyline, and GeometryMPolygon, are added to the collection in order to accommodate the spatial semantics when generating geospatial notifications (Figure 3). The geometries of simple geographic features can cover a large variety of the spatial component representation in generating geospatial notifications. Other types of data can be part of the geospatial notifications data contents, specifically computer files, including: images, documents, and media files.

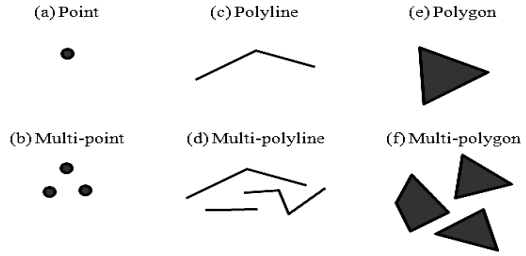

Figure 3: Geometrical representations of simple geographic features

Name/value pairs with byte-array data types are employed for this regard (Table 1). Byte-array data types handle any arbitrary information in binary format. Therefore, the required files are converted first to binary data then assigned to byte-array name/value pairs.

3.2 Geospatial Subscription Data Model

Generally, the design of the subscription model should follow the underlying notification data model. Two types of predicates are supported in the geospatial subscription language model: attribute predicates (AP) and spatial predicates (SP). Attribute predicates are utilized to constrain the selection of geospatial notifications according to their content descriptive data.

Table1: Examples of name/value pair formatting with a geometry data type

Geometry Type	Name/Value Format
Point, Multi-point	{GeometryPoint, (10 10)}, {GeometryMPoint, (10 10); (15 10); (13 12)}
Polyline, Multi-polyline	{GeometryPolyline, (10 10, 12 10, 15 12)}, {GeometryMPolyline, (10 10, 12 10, 15 12); (12 10, 12 15); (11 11, 8 10, 13 12)}
Polygon Multi-polygon	{GeometryPolygon, (12 14, 10 10, 13 15)}, {GeometryMPolygon, (12 14, 10 10, 13 15); (13 11, 10 15, 16 18); (11 12, 15 12, 12 14, 9 8)}

Geospatial subscriptions can be seen as filters or boolean-valued functions that evaluate whether or not published geospatial notifications match the defined constraints. Spatial predicates are introduced in the subscription language model offering the subscribers more expressiveness to define their interests in geospatial notifications that satisfy certain spatial constraints. SP is defined as triple parameters: base geometry G_p, a spatial operator *SOp*, and a buffer value *buff*, i.e. $SP_i = (G_{pi}, SOp_i, buff_i)$. The subscriber usually defines type, shape, and location of the base geometry by manual drawing on the screen or selecting existing geographic features using a GIS map. These functionalities are provided by the software application and allow subscribers to use. Table 2 shows some examples of spatial types of interests in geospatial notifications and their respective expressions of spatial predicates.

Table 2: Examples of spatial interests' expressions and their respective spatial predicates formation

Spatial Interest	Geospatial Notification		Respective Spatial Predicate		
	Source Desc.	Comparison Geometry	Base Geometry *Gb*	Spatial Operator *SOp*	Buffer Value *buff* (m)
Notify me of any Coast Guard Boat is within a coastal boundary	Boats current positions	Point	Polygon	Contain	0
Notify me of any oil spill incident happens within 5km off the coastal boundary	Oil spill incidents reporters	Point	Point	Contain	5000
Notify me of any oil spill spreading exists in within a coastal boundary	Temporal oil spill spreading area	Polygon	Polygon	Overlap	0
Notify me of any Coast Guard Boat far from the accident area by 2km	Coast Guard Boat current positions	Point	Polyline	Disjoint	2000

3.3 Geospatial Notification Matching

Notification matching is a prominent process executed by the notification service. The results from this process determine the flow of information between the interacting clients. The matching process determines a subset of subscriptions of *SP and* AP_i (*i.e.,* $Sub_i[SP, AP_i]$), where each Sub_i in the subset

matches the geospatial notification n_g (spatial constraint and an attribute constraint respectively). Both functions take the geospatial notification n_g as input, evaluate n_g according to the assigned conditions, and generate a boolean value as an output.
The Sub_i matches n_g if the output boolean values from both functions are *true* (*i.e., Sub_i[true, true]*), whereas the Sub_i does not match n_g otherwise (*i.e., Sub_i[true, false]*, *Sub_i[false, true]*, or *Sub_i[false, false]*). In cases where the key word *"NULL"* is assigned to any one of the subscription's functions, the output of the associated function is considered *true* without evaluating n_g.
Geospatial notifications matching in the context of the geospatial-based publish/subscribe can be achieved by spatial data indexing to enhance spatial data query processing. In this approach the geospatial data structure processed geospatial subscriptions into appropriate data structure to allow fast matching. In general, databases rely on the index data structure for quick access of data requested by a certain query, and that is in contrast with the traditional way of sequentially scanning the data entries which is considered a time-consuming and expensive process. Spatial indexing enhances the processing of spatial queries and speed up retrieving the data of the required spatial objects. The fundamental concept of spatial indexing is the use of approximations (Shekhar and Chawal 2003). Using a spatial index, the processing of an operation that involves a spatial predicate on a collection of spatial objects is performed in two steps: the filter step; selecting all the spatial objects whose mbb satisfies the spatial predicate. This step returns a superset of candidates of spatial objects. In the second step, called the refinement step, the exact geometries of the spatial objects in the superset are tested against the spatial predicate. This key procedure is behind querying and retrieving spatial data quickly and efficiently.
The matching engine initially pre-processes geospatial subscriptions into a data structure that allows fast matching. At a later stage and when a geospatial notification is published, the matching engine uses the prepared data structure of geospatial subscriptions and conducts the matching process searching for the matched ones. At the pre-processing stage, geospatial subscriptions are pre-processed into homogeneous feature classes and structured in spatial indexes. In the matching phase, as soon as a geospatial notification n_g reaches the notification service, the matching engine uses the prepared spatial indexes of registered geospatial subscriptions and executes the matching process. The process takes the geometry (i.e., the spatial component) of the geospatial notification and uses it as a spatial query to retrieve the matched set of geospatial subscriptions from the stored feature classes. As there are potentially six subscription feature classes, six spatial queries are executed. The formulations of these queries are as follows:

- (Geometry of n_g) Within (the geometries of Contain feature class).
- (Geometry of n_g) Contains (the geometries of Within feature class).
- (Geometry of n_g) Disjoints (the geometries of Disjoint feature class).
- (Geometry of n_g) Crosses (the geometries of Cross feature class).
- (Geometry of n_g) Touches (the geometries of Touch feature class).
- (Geometry of n_g) Overlaps (the geometries of Overlap feature class).

In the first two feature classes, Contain and Within feature classes, the spatial operators Within and Contains are used respectively as the spatial relationship is reversed by using the geospatial notification n_g as a query. Whereas, in the remaining four feature classes, Disjoint, Cross, Touch, and Overlap, similar spatial relationship, Disjoints, Crosses, Touches, and Overlaps are used in their respective queries as the function criteria does not change with reversing the base geometry and the comparison geometry (Figure 4). Lastly, the geospatial subscription features selected by executing the above spatial queries are considered matches to the geospatial notification n_g.

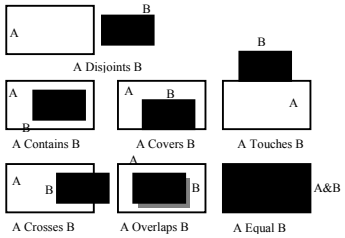

Figure 4: Six subscription feature classes cover above spatial data topological relationships

This matching approach is implemented by the matching engine of the notification service component to accelerate the dispatching process of published geospatial notifications.

3.4 Delivery of Geospatial Notifications

The middleware service matches published geospatial notifications with registered geospatial subscriptions and consequently finds the interested subscriber clients whose subscriptions are satisfied. The next stage is to deliver or push the information to the interested subscribers' applications. Technically, the notification service executes the output operation Notify (n_g) encapsulating the published geospatial notification within this operation and addressing the matched subscribers on the delivery process.
As the received geospatial notifications contain geospatial data, subscribers generally need to interactively visualize the received notifications on a GIS map and perform various spatial analyses based on the received data, such as overlaying, proximity, and network analyses, or potentially conducting responsive processes and actions upon receiving particular types of geospatial notifications. To this end, there are important issues that the interacting components should be aware of regarding effective handling of delivered geospatial notifications. First and foremost, the content-data schema of potentially published geospatial notifications should be well-known by the interested subscribers (i.e. receivers' applications). In this manner, the core processing of the subscribers applications can be customized in a way to handle, parse, and manipulate the received notifications data according to the requirements. For instance, a user requires overlying the received points' notifications of assets' current positions (Table 3) automatically in a GIS map and with a certain symbology style. Another user needs an automatic storage of the received notifications of oil spill remote sensors' as well as sea and weather conditions including currents and wind observations in a certain format inside a database. Second, the coordinate referencing system by which the spatial component of geospatial notifications is created should be known, too. The subscribers then can georeference the received geospatial notifications and perform an appropriate coordinate transformation processing required to be consistent with the coordinate system of their GIS datasets.
Content based data structures, is assumed in the implementation of the real-time oil spill emergency response system which adopting the advertisement operations in the geospatial-based publish/subscribe model.

4. OIL SPILL MONITORING USING (SLEAF) SENSOR

Laser Fluorosensors, such as the Scanning Laser Environmental Airborne Fluorosensor (SLEAF) sensor operated by Environment Canada, are among the most appropriate sensors for oil spill surveillance in light of their ability to detect oil on

all backgrounds, classify different types of oil, and detect oil on shorelines. Real time laser fluorosensor data processing constitutes one of the most useful monitoring systems for oil spill detection and decision support. Certain aromatic hydrocarbon compounds in petroleum oils absorb laser-induced UV light to become electronically excited. The excitation is released through fluorescence emission by the compound mainly in the visible region. A multi-channel receiver can be used to record the fluorescence spectrum (Goodman, 1994).

Table 3: An example of content data structure for geospatial notification of assets' current positions

Attribute Name	Data Type	Description	Example
BoatID	String	ID of the Oil Spill Responder group	{BoatID, "A1"}
X	Double	X-axis coordinate (UTM Zone 11N)	{X, -1211956}
Y	Double	Y-axis coordinate (UTM Zone 11N)	{Y, 1535061}
ObsTime	String	Current observation time	{ObsTime, "24/11/2008 15:12:00"}

Fluorescence spectrum of gelbstoff and phytoplankton look different from that of petroleum oils. Moreover, different types of oils have distinct fluorescence emission signature which allows for reliable oil identification. Oils can be classified also on the basis of fluorescence decay time (Goodman, 1994). The energy transfer between incident light and water molecules is known as Raman scatteringwhich is useful for fluorescence calibration as well as for estimating oil thickness to some extent (Brown and Fingas, 2003). The detail about methodology used for oil spill SLEAF image processing and the system functionality for monitoring oil spill including trajectory modelling and output data visualisation in Web GIS is presented in (Jha, 2008).

5. A PUBLISH/SUBSCRIBE BASED OIL SPILL MONITORING AND MANAGEMENT SYSTEM

An oil spill monitoring and management system prototype has been developed based on the geospatial publish/subscribe interaction model. The main three components of the system are publishers, subscribers, and the notification service middleware (Figure 5). The system architecture is divided into a three-tier system model: the client tier, the business logic tier (also called application server), and the database tier. Publishers and subscribers fall inside the client tier. They interact with the middleware tier by sending and receiving messages (*i.e.,* geospatial events/notifications). The notification service component resides in the business logic tier, where most of the application processing work occurs. The business logic tier, from one side, communicates with the client tier by handling all the incoming publications and subscriptions and consequently sending out notification messages. On the other hand, the processes running inside the business logic tier are permitted access to the database tier for data storing and retrieving as well as SLEAF data processing. A portion of the business logic functions resides in the users' applications in the client tier for processing and visualizing the messages data. APIs and TCP/IP protocols facilitate the underlying communication among the tiers. Higher level system architecture for oil spill monitoring and management system is presented in Figure 6. The key components include the user's layers, the application data processing layer and the communication layer. Users submit remote sensing (SLEAF) data to the system and the processed results automatically pushed to relevant end users in a real-time manner.

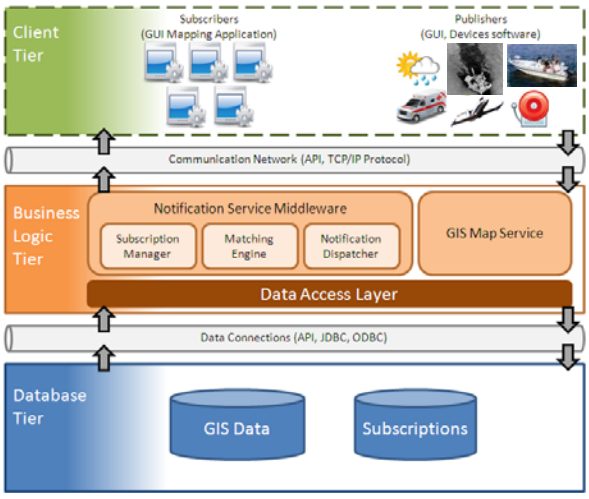

Figure 5: The main three components of the prototype system in publish/subscribe interaction model

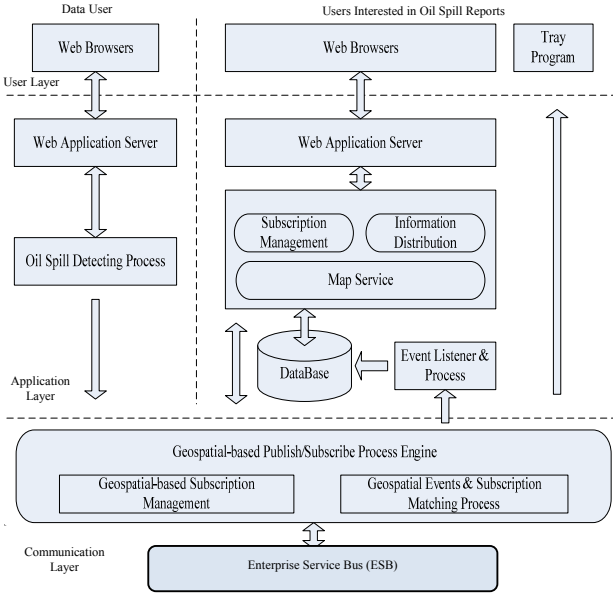

Figure 6: High-level architecture of oil spill detection and information distribution system

The system is a web-based application (Figure 7). Data users can access the system and submit SLEAF data. Methods of oil spill detection and geo-reference are realized in oil spill detecting process module. The module automatically geo-references and processes SLEAF data (Jha, 2008), vector features from oil spill locations are extracted and the results are published to the communication layer in XML format. For the public users a WebGIS application is developed using ArcGIS Server V9.3 and Ajax. These applications provide functionalities of subscription management and information distribution. Users can submit their subscriptions (based on a geospatial location of interest or an oil spill attribute constraint). All geospatial-based subscriptions are maintained and managed in the geospatial-based publish/subscribe process engine. When an oil spill is detected the information will be pushed to the relevant subscribers and a notification will flash in their webpage so that they can view and act on the detailed information. The function of the communication layer is to

convey events information (messages) to the correct consumers of information.

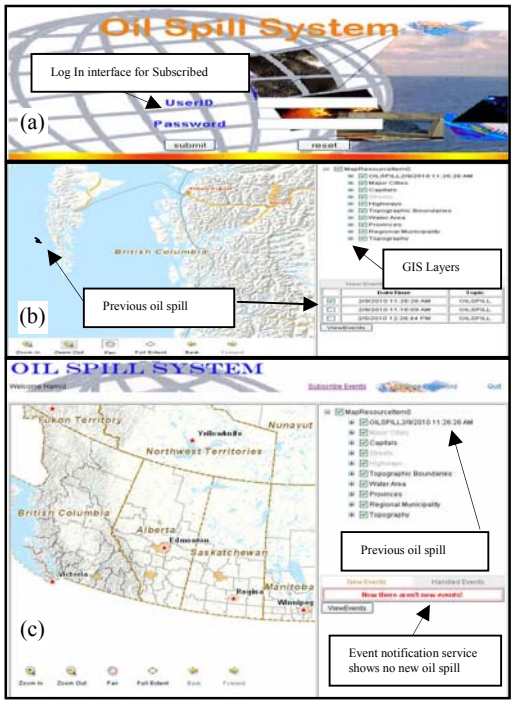

Figure 7: Log in interface directs subscriber(s) to the main oil spill monitoring and management system (a); and notify new events in real time (b) immediately after new data has linked from SLEAF processing engine for oil spill data analysis (c).

The communication layer is composed of a geospatial-based publish/subscribe process engine (GPSPE) and an Enterprise Service Bus (ESB) (Kassab, 2009). The Enterprise TIBCO Messaging Service (EMS), a Topic-based event-driven message middleware, is adopted as the ESB in this system. The TIBCO EMS middleware receives published events and pushes them to consumers. The GPSPE (an extension for TIBCO EMS) implements the geospatial-based publish/subscribe model proposed in this research: it is developed based on ArcSDE 9.3 and Microsoft SQL Server 2005. Corresponding to the six elements in Figure 5, the GPSPE defines six feature classes to store all geometry objects in geospatial-based subscriptions. Spatial indexes are separately constructed by ArcSDE. GPSPE carries out three jobs. First it receives oil spill events and extracts the object geometry (which describes the location or area of the oil spill). Second, it uses a spatial query to match the events and subscriptions. Third, it publishes the matching results to the ESB. The event listener process module carries out four activities: it listens to the ESB; it receives the oil spill events; it matches the results; and finally it converts them to the appropriate format and stores them into the database. The information distribution module obtains the information from the database and distributes oil spill information to users when they are online (or prompts to them the first time they login). In this way it is guaranteed that users do not miss any subscribed oil spill information.

Conclusion

Such a system will to help provide a real-time system that can assimilate ocean data into numerical computer models that simulate ocean dynamics and generate forecasts of ocean conditions. This information is useful for fishers, coastal managers, scientists, and disaster professionals. It is suggested that future research consider other types of remote sensing data like radar satellites and modern trajectory simulations for oil spill forecasting.

References

Brown, C., Fingas, M., 2003. Review of the development of laser fluorosensors for oil spill application. *Marine Pollution Bulletin*, 47, 477–484.

Cao, F., 2006. *Architecture Design for Distributed Content-Based Publish-Subscribe Systems*, unpublished thesis Princeton University.

Eugster, P. T., Felber, P. A., Guerraoui, R. and Kermarrec A. M., 2003. The Many Faces of Publish/Subscribe. *ACM Computing Surveys (CSUR)*, 35, pp. 114 - 131.

Goodman, R., 1994. Overview and Future Trends in Oil Spill Remote Sensing. *Spill Science & Technology Bulletin*, 1.1, pp. 11-21.

Graham, S., Niblett, R., Chappell, D., Lewis, A., Nagaratnam, N., Parikh, J., Patil, S., Samdarshi, S., Sedukhin, I., Snelling, D., Tuecke, S., Vambenepe, W. and Weihl, B., 2004. "Publish-Subscribe Notification for Web services, Version 1.0," http://www.oasis-open.org (accessed 2010).

Harrison, M., 1995. *The USENET handbook: a user's guide to Netnews,* O'Reilly & Associates, Inc. Sebastopol, CA, USA.

Holloway, S., 2008. "Are you ready for M3O-Vitria's new BPM offering". http://www.vitria.com (accessed 2008).

Jha, M.N. Levy, J. and Gao, Y., 2008. Advances in Remote Sensing for Oil Spill Disaster Management: State-of-the-Art Sensors Technology for Oil Spill Surveillance. *Sensors*, 8, pp. 236-255.

Kassab, A.S., 2009. *Geospatial-based Publish/Subscribe: Improving Real-time Notification and Situational Awareness in Fire Emergency*, MSc. Thesis, University of Calgary, Canada.

Keramitsoglou, I., Kiranoudis, C. T., Sarimveis, H. and Sifakis, N., 2004. A Multidisciplinary Decision Support System for Forest Fire Crisis Management. *Environmental Management*, 33 (2), pp. 212-225.

Muhl, G., 2001. *Generic Constraints for Content-Based Publish/Subscribe,* translated by Trento, Italy: Springer-Verlag, London, UK, pp. 211 - 225.

Muhl, G., Fiege, L. and Pietzuch, P., 2006. *Distributed Event-Based Systems.* Springer, Germany.

Oki, B., Pfluegl, M., Siegel, A. and Skeen, D., 1994. *The Information Bus - An Architecture for Extensible Distributed Systems,* translated by Asheville, North Carolina, United States: ACM New York, USA, pp. 58 - 68.

Shekhar, S. and Chawla, S., 2003. *Spatial Databases: A Tour,* Prentice Hall.

TIBCO. 1999. *TIB/Rendezvous.*

TIBCO., 2000. "TIBCO - Service Oriented Architecture (SOA) Software, Business Process Management (BPM) Software Leader". http://www.tibco.com (accessed 2008).

Acknowledgments

The financial support for this research from GEOIDE (Canadian Geomatics for Informed Decisions) NCE (Networks of Centers of Excellence) and NSERC (Natural Sciences and Engineering Research Council) of Canada is greatly acknowledged.