Approach for discovering and handling crisis in a service-oriented environment

Nabil Adam, Vandana P. Janeja, Aabhas V. Paliwal, Basit Shafiq, Cedric Ulmer, Volker Gersabeck, Anne Hardy, Christof Bornhoevd and Joachim Schaper

***Abstract*—In an emergency situation failure to respond in a timely manner poses a significant threat. Data needed for timely response comes from various sources and sensors. These individual data streams when viewed in isolation may appear irrelevant, however, when analyzed collectively may identify potential threats. An effective and timely response also requires collaboration and information sharing among various government agencies at all levels. This collaboration information sharing among agencies can be achieved using service-oriented architecture, where agencies provide access to their information resources and applications using Web services. Each of these agencies has its own rules/policies for providing their services. It is therefore, important to verify the correctness of the emergency response processes with respect to the rules/policies of the collaborating agencies involved in the execution of such processes. In this paper we present an approach which addresses the above challenges. Specifically, the proposed approach: a) employs multi stream data mining for identification of potential threats and disambiguation of alarms; b) provides a methodology for the discovery and selection of relevant Web services; c) employs a timed automata based verification methodology for determining the correctness of emergency response processes with respect to the rules of the collaborating agencies. We provide an overview of the initial implementation of the proposed approach.**

***Index Terms*— Emergency management and response, Event analysis, Process verification, Web service discovery and selection.**

I. Introduction

In the context of public security, one of the key challenges is achieving effective, timely and systematic collaboration and information sharing among various government agencies at the Federal, state, and local levels. In case of a crisis such as a natural disaster, a virtual response team needs to be formed in an ad-hoc manner. Members of this virtual response team come from various government agencies and private organizations. We call this as the Virtual Multi-Agency Response Team (VMART) [4]**.** Depending on several factors, including the location and nature of the crisis, the composition of the VMART may change from one crisis to another. Furthermore, during the course of a given crisis the membership in the VMART may also change dynamically to accommodate various needs (e.g., public health versus fire) and to conform to certain constraints, such as jurisdictions, e.g., the crisis extends, initially from New York to New Jersey.

Collaboration among the VMART agencies includes sharing of relevant information that helps these agencies to discharge their duties and fulfill their responsibilities within the overall effort. The information and data services provided by the collaborating agencies may be composed to realize emergency response operations that require interoperation among various agencies in a coordinated manner. This interoperation and information sharing among agencies can be achieved using service-oriented architecture, where the agencies provide access to their information resources and applications using Web services [2]. Access to such Web services is governed by certain rules and regulations, which are specified by the corresponding service provider agencies. In addition, appropriate agencies in the VMART environment specify rules and policies for composition and activation of response operations to deal with the emergent situation in the form of a service flow. However, given the increasing semantic heterogeneity in the functional specifications and policies of Web services, it is important to discover and select the appropriate services that can provide the desired functionality specified in the service flow. Moreover, the service flow specifications need to be analyzed for conformance with the service policies of corresponding agencies.

Another important factor to be considered for effective emergency management and response is that the data comes from various sources and sensors. In addition, certain events when viewed in isolation may appear irrelevant or benign however, when analyzed collectively may identify a potential threat. Therefore a key challenge, for identifying critical events that pose a threat, is to evaluate individual events and events in combination.

For the purpose of illustrating these issues we provide the following scenario.

Manuscript received January 8, 2007. This work is supported in part by the National Science Foundation under grant IIS-0306838 and SAP Labs, LLC.

Nabil Adam, Vandana P. Janeja, Aabhas V. Paliwal and Basit Shafiq are with CIMIC, Rutgers University, Newark, NJ, 07102, USA (973-353-5239; fax: 973-353-5808; (e-mail: adam@cimic.rutgers.edu).

Cedric Ulmer, Volker Gersabeck Anne Hardy, Christof Bornhoevd and Joachim Schaper are with SAP Labs, LLC, Palo Alto Research Center, Palo Alto, CA 94304, USA (e-mail: cedric.ulmer@sap.com).

A. Motivating Scenario

Consider a scenario where a major hurricane is expected to make landfall in a major metropolitan area of North Carolina State (NC). For safety of the general public, the Governor of the State declares an emergency situation and issues a mandatory evacuation order for the residents of the metropolitan area. Furthermore, in order to facilitate evacuation of residents in timely manner traffic Reversal Control Group (RCG) is formed to analyze the possibility of reversing the lanes of major highways in the metropolitan area. The RCG is made up of several agencies including, the State Department of Transportation (NCDOT), the State Highway Patrol, and the Division of Emergency Management [22]. For making such strategic decision, RCG continuously monitors the National Hurricane Center advisory and the forecast of the storm. RCG also takes into consideration additional factors including, the time, in which evacuation needs to be completed, the numbers of residents to be evacuated, the current traffic conditions, the availability of resources and staff to manage contra-flow operations. Once RCG takes the decision for lane reversal, it coordinates with various local and state agencies for execution of the lane reversal process. For monitoring of such conditions, RCG may employ third party services such as real-time traffic video services, video analysis services, weather advisory services, and news reports.

B. Focus of the Paper

As evidenced by the above scenario, there is a need for the RCG to analyze events (e.g., wind speed, change in projected path, and traffic congestion) collectively based on spatial, temporal, and other relevant attributes rather than individually. Thus, being able to identify threats and critical events from low level or incomplete alarm information. Each alarm may report some aspects of the same or related events, which may not be apparent when these alarms are analyzed individually. The sheer number of alarms and alerts from different sensors and sources may overwhelm the emergency responders. Moreover, some of these alarms and alerts are redundant or false positives. In section III we present our proposed approach for joint analysis and correlation of events/alarms and identification of false positives.

The above scenario also illustrates the application of service-oriented architecture and the extensive use of Web services. The need for semantic-based, rather than keyword-based, discovery and composition of services is evident since this would results in increasing the likelihood of retrieval of relevant services with minimum human intervention and in shorter time. The discussion in section IV addresses the issue on Semantic based Service Discovery and Selection.

Finally, as the above scenario illustrates, the emergency response process involves interoperation among multiple agencies in terms of providing the required services (or execution of designated tasks) in a coordinated manner. Each of these agencies has its own rules/policies for providing their services and these rules/policies may incorporate temporal dependencies and real-time constraints. It is therefore, important to verify the correctness of the emergency response processes with respect to the rules/policies of the collaborating agencies involved in the execution of such processes. We address the issue of policy/rule verification in such distributed environment in section IV.

Fig 1 illustrates the framework for our approach. It consists of three interoperating components, (a) Event Analysis, (b) Semantic based Service Discovery and (c) Verification. The *Event Analysis* component, consisting of event prediction and alarm clustering, is formed by the Event Listener that feeds into the Event Analysis sub-component. The Event Analysis component outputs to the Control Unit which interacts with the Event/Engine Mappings module. The *Semantic based Service Discovery* component, facilitating the discovery, of appropriate services consists of Action Performer, Actions' DB, Web Service Discovery and the UDDI sub-component. The interoperations of the subcomponents that form Semantic Service Discovery component are illustrated in Fig 6. The *Verification* component comprises the Rule Engine, Persistence Manager, Policies' Repository and the Knowledge Base Editor. It verifies the correctness of the emergency response processes with respect to the rules of the collaborating agencies. Due to lack of space we do not provide the details of the interoperations within/between the components of the framework in this paper. An initial implementation of this framework is described in section V. Below is a description of the event analysis, semantic based service discovery, and verification components of Fig 1.

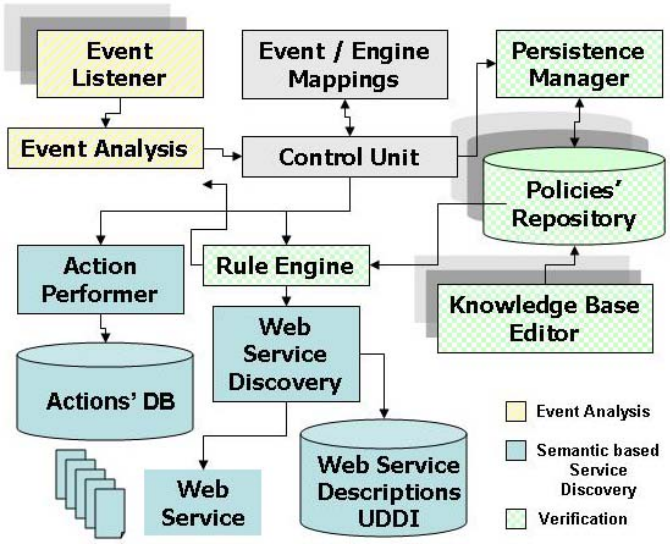

Fig. 1.Overall Architecture

II. EVENT ANALYSIS

In an emergency management situation, information needed for effective and timely response comes from various sources and sensors. For example, in the evacuation scenario described in Section I, RCG receives information and event reports from field officers, subscribed Web services (e.g.,

weather advisory services, traffic surveillance services), and various sensor devices such as traffic sensors, video surveillance cameras, and spill detection sensors. The data and alarm messages coming from the different sensors and sources can be viewed as time series. These multiple time series can be analyzed collectively for identification of potential threat conditions and detection of false positives. Secondly there is a lot of redundancy across the time series data since several sensors record the same event. Thus there is a need for disambiguation of the alarms being generated. In order to address these two issues we propose to focus on two aspects of event analysis *(a) Event Prediction:* to process data streams in a timely manner and for time series prediction based upon the past values, *(b) Alarm Clustering:* for disambiguation and correlation of alarms. We next discuss the two issues in detail.

A. Event Prediction

In an emergency response situation, failure to respond in a timely manner poses a significant threat. This threat may result from various correlated events. For example, in the hurricane scenario, some of the correlated events include deteriorating traffic situation because of accidents resulting in lane closure, bad weather conditions, and flooding of roads. An early detection of a potential threat may help the decision makers to take appropriate actions to deal with the emergent situation in a timely manner. As discussed above, the traffic conditions and weather parameters change with time and can be viewed as evolving time series. In this respect, a potential threat or critical event may be detected based on the alarming values of one or more of the time series in some time interval. This alarming value of the time series can be predicted using the past values of the same series and the past and present values of other correlated time series [15, 25, 28, 20, 21].

In the event of a crisis, massive amount of data is received from numerous sensors and sources at a very high rate. Therefore processing of such data streams in a timely manner is a challenging task and there is a need for efficient stream mining algorithms with low computation complexity. Another major requirement for time series prediction is the incremental and online computation of future values based upon the portions of each time series data seen so far, without waiting for their completion as such data streams may be indefinitely long and may not have a predictable termination point [28].

For event prediction and threat detection using time series analysis, we employ the online and incremental data mining approach for pattern discovery in multiple time series proposed by Papadimitriou *et. al.* [25]. This approach relies on principle component analysis for determining the hidden variables and principle components. Thus the key trends in the entire collection of the time series is summarized. This summarized information can be used to reconstruct the original time series values. The reconstructed values of the time series can be computed by taking the vector product of the hidden variables and the principal components. The difference between the reconstructed value and the original time series value is called the reconstruction error. The number of the hidden variables and principle components used for summarization of the time series data depends on the degree of correlation across these series. A higher degree of correlation implies that lesser number of hidden variables and principle components are needed for reconstruction of the original time series with small reconstruction error. Since in an emergency situation, the time series data received from different sensors and sources is not completely independent, principal component analysis provides an efficient way to process the time series data. Specifically, the principal component analysis can help to do fast prediction of time series values for event/threat detection. Additionally, this analysis can be utilized to identify correlations between various time series from various data sources such as the sensor data sources depicted in our motivating example.

The method for computation of the principle components and hidden variables proposed by Papadimitriou *et. al.* in [25] is efficient and it satisfies the timely processing requirements of the data streams discussed above. The computational complexity of this method scales linearly with the number of data streams is independent of the size of each stream. Moreover it is incremental and can be used to detect any change in the hidden variables and principle components based on the current and past values of the multiple data streams without waiting for their completion. This is important as the sensor data values keep on changing during an emergency situation and accordingly the principle components and hidden variables need to be adapted dynamically to summarize the current data trends and behavior.

Since the hidden variables and the principle components provide a compact representation of the original time series and guarantee a high degree of reconstruction accuracy, we can apply the value prediction algorithm such as multi-variate linear regressive algorithm [20] to predict the values of hidden variables and then use these predicted values of hidden variables to reconstruct the original time series at future instants. The multi-variate regressive algorithm essentially computes the future values of data based on the past and present values. This regression analysis can be directly applied on the original time series data to predict the future values [28]. However, applying regression analysis on the hidden variables and using them to predict the time series values will have a significantly lower computational complexity than directly applying regression analysis on original time series. The reason for this reduced complexity is that typically the number of hidden variables is much smaller than the number of time series. Consequently, a lesser number of auto regressive coefficients need to be determined for the hidden variable case [25].

B. Alarm Clustering

The sensors and incident reporting sources often generate a large volume of low-level and incomplete alarm information because of the large number of activities and conditions being monitored in the emergency management situation. Often the alarms and incident information reported by different sensors and sources correspond to the same event. The large volume of alarms and reports by different sensors and sources may prevent the decision makers from making an accurate assessment of the emergency situation and initiating appropriate response in a timely manner. For improved situational awareness and effective response, it is important to reduce redundant alarms, consolidate and correlate multiple alarms that are independently generated by local sensors but may point to a common incident. For disambiguation and correlation of alarms, we propose a data clustering-based approach.

Our proposed approach for alarm disambiguation and event correlation consolidates different alarms and events based on their spatial, temporal proximity, semantic relationships, and the similarity of the attributes and the corresponding time series generating these alarms. Specifically, the clustering of related events is based on the following criteria:

- Alarms generated by sensors that are within close spatial and temporal proximity are more likely to indicate similar aspects of the same or related events.
- Two or more alarms falling in the same cluster are likely to have similar characteristics [18]. This is due to the way a cluster is formed based on computing a similarity such as distance between the attributes of events.
- In addition similarity of events may be determined using semantic relationships between the attributes of two or more alarms, due to which alarms are likely to be correlated and may fall in the same cluster. Here semantic relationship among the attributes of different entities refers to an ontology based relationship between relevant attributes of events [3, 19].

Let us consider the traffic slow down situation on the designated evacuation route opened for contra-flow traffic during evacuation of a metropolitan area. This situation is depicted in Fig 2. The traffic slow down may occur at different locations because of lane closures (due to accidents, falling debris, vehicle stall etc.), road flooding, or bad weather conditions. Various traffic and road sensors installed at different locations may report the traffic slow down within their localized monitoring area. Fig 2 depicts the time series corresponding to the traffic flow and wind speed monitored at different time instances by sensors installed at the entry ramps to the highway designated as the evacuation route. Since these traffic ramps are within the same metropolitan area, therefore the wind speed profile is similar across these locations as weather conditions do not change drastically across different observation points that are in close spatial proximity. Also, in the event of evacuation, it is expected that the traffic flow rate at all the entry ramps will be close to their maximum capacities. However due to the varying conditions on road the traffic flow profile may vary. In Fig 2, the traffic flow at the entry ramps B and C significantly decreases with long queues of vehicles. The local sensors at these locations trigger separate alarm due to the deteriorating traffic situation, which

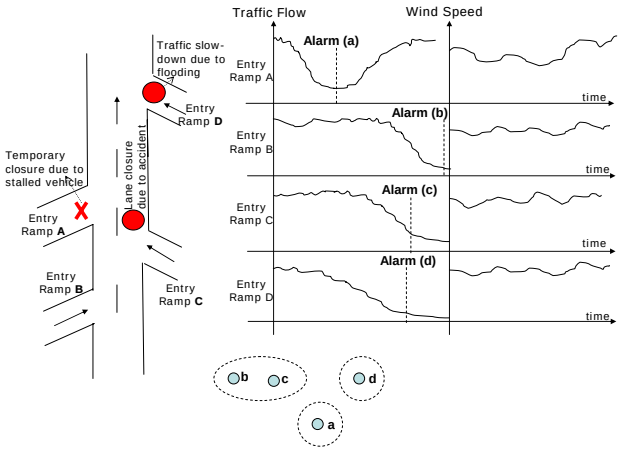

Fig. 2. Example of traffic slow down situation on a designated evacuation route due to various traffic incidents.

is caused by the accident at the highway. A traffic slow down alarm is also triggered at entry ramp D due to road flooding. Similarly a temporary closure of one of the lanes is shown due to a stalled vehicle at entry ramp A generates a traffic slow-down alarm. However, the traffic flow at ramp A becomes normal after removing the stalled vehicle from the blocked lane.

In order to determine the grouping of related alarms, we need to consider the proximity of these alarms in all dimensions including spatial and temporal dimensions, semantic dependence, and similarity in the characteristics of relevant attributes. In the example of Fig 2, the wind speed profile is similar for each of the four alarms as discussed above. However, the traffic flow time series associated with alarm ‘a’ has a low degree of correlation with the traffic flow time series corresponding to alarms ‘b’, ‘c’, and ‘d’. Based on this dissimilarity in the traffic flow series attribute alarm ‘a’ cannot be included in the clusters containing alarms ‘b’, ‘c’, or‘d’.

The traffic flow time series for alarms ‘b’, ‘c’, and‘d’ have a high degree of correlation. However, alarm ‘d’ is not included in the cluster containing alarms ‘b’ and ‘c’ despite the close proximity of alarm d with b and c in all the attributes. The reason for this separate clustering of alarms is the dissimilarity of the traffic flow time series at entry ramp A with the traffic flow series at ramps B, C, and D. Since, ramp A is between ramp D and ramps B and C. Therefore if traffic slow-down at ramps B, C, and D is caused by the same event, it should also affect the traffic at ramp A in a similar manner. Since the traffic profile at ramp A is significantly different from the traffic profiles at ramps B, C, and D, the traffic slow down at ramp D and ramps B and C can be attributed to two different events and therefore the corresponding alarms cannot be consolidated. The semantic dependence of clustered alarms is determined based on the attribute correlation of the event points in the spatial neighborhood of alarms clustered

together. Here Spatial Neighborhood refers to a group of objects which are in a spatial proximity. These objects in a spatial neighborhood are governed by the property of spatial autocorrelation where nearby objects behave similarly. The discovery of spatial neighborhood [1] itself is a significant aspect of any spatial analysis. Thus the semantic dependence is determined by the attribute correlation of events in the spatial neighborhood. The semantic dependence between attributes may be specified as a rule in the knowledge base (such as alarms generated by two different sensors installed at different locations with a high degree of similarity in all attributes cannot be clustered together if a sensor at some intermediate location exhibits a different behavior in one or more attributes). This semantic dependence may also be expressed as semantic relationship ontology between the attributes of different entities. This ontology based representation of semantic relationship between related events has been considered in one of our earlier work in which we utilize the ontological evaluation of known chemical combinations which result in an explosive combinations [19].

We focus on the attributes and their proximity measures for clustering of related alarms. Any data clustering approach can be used provided that the clustering criteria and the proximity measures described above are used for clustering of related alarms. As such clustering approaches primarily include partition based, hierarchical, grid based, model based and density-based techniques. Among these, density based clustering is more relevant to our work since it addresses spatial clustering. These approaches (e.g., DBSCAN [13] and OPTICS [5]) utilize the density-based notion of clustering i.e. they aggregate the data based on the sparsity or density in various regions of the data. DBSCAN(Density Based Spatial Clustering of Applications with Noise) utilizes the neighborhood of a point which contains a minimum number of points that are at a distance smaller or equal to the minimum radius determined as an input parameter. OPTICS (Ordering Points to identify the clustering Structure) enhances this approach to addresses the issue of traditional clustering as well as the intrinsic clustering structures for which they bring into the process local parameters for each cluster. Moreover, it is less sensitive to parameters as compared to DBSCAN. We skip the details of our alarm clustering algorithm because of space limitations.

For alarm clustering, the spatial and temporal proximity of alarms can be easily determined based on the location of the sensors generating the alarms and the time instants at which these alarms were generated. However, determining the proximity with respect to other attributes may not be straightforward as some of the attributes may have time varying characteristics e.g., traffic flow, weather conditions. The proximity of such attributes across different alarms cannot be established based on their current values. We employ multi-time series analysis to determine the similarity of attributes of similar types across different alarms. In particular, we use the pattern discovery approach for co-evolving time sequences to determine the similarity among the attribute time series [25]. We next discuss the semantics based web service discovery for an emergency scenario.

III. Semantic Based Web Service Discovery

Service requestors, within the VMART, have access to a choice of descriptions to various services that provide similar service functionality. Most service descriptions that exist to date are of syntactic nature. Although the emerging service description paradigm has more associated semantics through ontologies, current service discovery approaches often adopt keyword-matching technologies to locate the published web services. This syntax-based matchmaking returns discovery results that may not match the service requestor's original request. The discovery process is constrained by its dependence on human intervention for choosing the appropriate service based on its semantics. Automation of dynamic web service discovery is made viable by expression of domain semantics or domain specific knowledge [16]. In this paper we propose a methodology for the discovery of appropriate services that can provide the desired functionality for various crisis management tasks. We next describe our approach to semantic based web service discovery and selection.

Our semantic based web service methodology involves three key steps as presented in the discoverWebService algorithm illustrated in Fig 3. The key steps include 1) Ontology guided Service Categorization, 2) Web Service Parameter Association Mining, and 3) Service Request Expansion. The details of our methodology are described below.

1) Ontology guided Service Categorization

For step 1, our service discovery methodology proposes an ontology guided web services categorization that takes advantage of hierarchical clustering and ranking semantic relationships. Our methodology builds on that described in [24] that does not assume term independence. The basic idea is to represent each web service as a vector that comprises of the terms of the service description and of the service's input and output parameters. We refer to this vector as the service description vector. Initially, we add, to the service description vector, relevant ontology concepts and/or delete, from the service description vector, irrelevant terms based on the ranking of semantic relationships among the ontology concepts. This is followed by pre-processing of the service vectors. Next, we apply hierarchical clustering to this web service dataset. We then associate an upper ontology concept for each cluster and the relevant ontology concept for the corresponding sub cluster. Following this, we retrieve the web service entries to represent the semantic information in the UDDI by creating tModels in the registry [26]. The tModel corresponds to concepts from the upper ontology representing functionality of the service in a relevant domain. The ontology is linked with the respective tModel using the *overviewURL*:tag of the tModels. Given the categorized set of web services based on features extracted from WSDL files, the

Algorithm: discoverWebService

Input: Web Service (WS), Web Service Request,
Ontology framework T = {T_0, …, T_n}; T_0: upper merged ontology
Output: Mapped Request Vector SRm, associated WSDLs

1: begin
2: /* Step 1. ontology guided service categorization */
3: for each web service vector $ws_i \in$WS do
4: Modify Web Service Vector (ws_m)
5: return ws_m
6: end for
7: hierarchically cluster WS to
WS clusters set φ = {φ1, φ2, …, φn}
8: for each web service cluster set φi do
9: associate upper concept cj⊆ T_0 to φi
10: end for
11: for each web service $ws_{m_k} \in \varphi_k$
12: Update $tModel_k$ to include c_φ
13: end for
14: /* Step 2. WS parameter associations mining */
15: form the association pattern itemset from WS cluster sets
16: perform Hyperclique pattern discoveries
for Association Pattern Collection (P)
17: for each association pattern p(x,y) $\in$ P do
18: rank semantic associations to compute $Score_{SemRank}$
19: return $Score_{SemRank}$
20: prune patterns with lower confidence level
21: /* a new set of patterns P' is created */
22: for each association pattern p(x,y) ∈ P' do
23: Search in WS for parameters p(x,y)
24: if p(x,y) covers parameters
25: select ws_i
26: end for
27: return ws_i
28: /* Step 3. WS parameter associations mining */
29: for web service request SR do
30: preprocess document WS_{ci}
33: generate enhanced service request vector
33: end for
34: return SR_e
35: project enhanced service request SRe to reduced k-space
36: calculate proximity using cosine measure for similarity
37: return SR_m
38: retrieve associated WSDLs
39: end

Fig. 3. Semantic based Web Service Discovery Algorithm

next step involves selection of web services based on parameter association mining.

2) Web Service Parameter Association Mining

In the next step, our discovery methodology takes advantage of association pattern mining and ranking semantic relationships. The basic idea of our approach is to represent parameters of the ontology guided categorized web services from step 1 as a vector in which each entry records the terms of the operations' input and output. Thus each of these collections of terms forms a transaction. The set of web services is represented by a collection of a set of transactions. Next we mine this web service collection to find the frequent hyperclique patterns that satisfy a given support level and h-confidence level [27]. This is followed by a pruning of the hyperclique patterns on the basis of the ranking of semantic relationships among the terms. Then for each remaining pattern we retrieve the web services that have the pattern expressed as part of the service description. The next step involves semantic service request expansion. The expanded service requests are then matched against the retrieved set of web services for appropriate service discovery.

3) Service Request Expansion

Step 3 of our methodology for web service discovery combines ontology linking and Latent Semantic Indexing (LSI), which is a statistical approach used to capture term relationships and underlying domain semantics [12]. The basic idea of our approach is to build the service request vector according to the domain ontology, have the training set of the LSI classifier based on features extracted from selected WSDL files, and finally project the description vectors and the request vector and utilize the cosine measure to determine similarities and to retrieve the corresponding relevant WSDL service descriptions.

IV. VERIFICATION

Emergency response processes are designed for repeated/recurrent activations meaning that whenever an emergency situation arises, the relevant response process needs to be activated in a timely manner. An emergency response process involves interoperation among multiple agencies in terms of providing the required services (or execution of designated tasks) in a coordinated manner. These agencies may be autonomous entities and have their own set of rules for provisioning of their services. Moreover, these rules may incorporate temporal dependencies and real-time constraints on provisioning of the respective services. Therefore, it is important to verify the correctness of the emergency response processes with respect to the rules, in the form of a rule set, of the collaborating agencies involved in the execution of such processes. In particular, the correctness verification of the process specification involves ensuring the following:

1. The rule set of collaborating agencies are non-conflicting with respect to the specifications of the emergency response process.
2. The rule set of collaborating agencies support execution of the emergency response process under the given timing constraints.

For verifying the above properties, we need a formal model for representation of the time-dependent rule sets of collaborating agencies and modeling of the process specifications. Various verification methodologies have been proposed in the area of Web service composition, distributed protocol synthesis, and distributed real-time systems [9, 10, 14, 6, 7]. These methodologies use various state-space models to represent the underlying specifications of different components. These components may correspond to Web services, protocol description, or interacting real-time systems. For verifying whether the overall behavior of the components exhibit certain desired/undesired properties, these methodologies employ model checking based approaches where the desired/undesired properties are either represented as a scenario [6, 7] or as a logic formula [9,10].

We use timed automata for specification of the rule sets of collaborating agencies. The reasons for selecting timed automata over other state-based models are given below:

- Timed automata provide a generic representation of the specification of rules with temporal constraints. These temporal constraints can also be specified using absolute time values, which is difficult to model in other state-based models including timed Petri nets [8], timed transition systems [23], and Modecharts [17].
- Standard methods for automata composition can be applied to integrate the timed automata corresponding to different components into a global automaton.

For correctness verification, we build on the scenario matching approach for model checking of real-time properties [7,6]. In this approach, any property that needs to be checked in a given system is specified using *visual timed event scenario* (VTS) diagram. The VTS specification also allows modeling of properties with real-time constraints. Given a scenario and system specifications modeled as timed automaton, the VTS-based scenario matching approach searches for a trace in the timed automaton that matches with the given scenario. A matching trace in the automata implies that the property corresponding to the scenario is supported by the system specifications. This scenario matching approach is based on the *existential* semantics meaning that matching of the scenario with at least one trace of the automaton is sufficient to say that the corresponding property may be exhibited by the system. However, there can be numerous traces that can be generated from a given automaton that may not match with the given scenario. The existential scenario matching approach is particularly useful when the system specifications need to be analyzed for absence of certain undesirable properties such as occurrence of forbidden events and violation of deadlines. In this case the undesirable properties are modeled as VTS diagrams and the timed automaton corresponding to the system specifications is searched for matching traces. The existence of any matching trace implies that the system may exhibit the undesirable property and therefore there is inconsistency in the system specification.

As VTS allows specification of events with temporal ordering and real-time constraints, the interactions among the agencies corresponding to an emergency response process can be represented as a VTS diagram. Fig 4(a) show a VTS

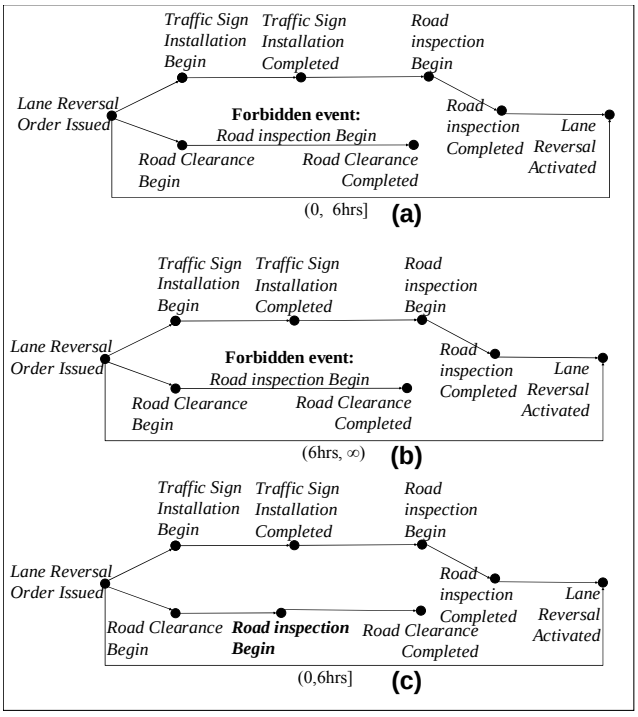

Fig. 4. VTS diagram of the lane reversal process scenario shown in (a). complimentary scenarios of (a) shown in (b),(c).

diagram corresponding to the lane reversal process. In the VTS diagram, an event is represented as a point and a directed edge between two points indicates the temporal ordering of the corresponding events. The optional time interval associated with an edge, specifies the lower and upper bounds on the temporal gap between the occurrences of events connected by the edge. For example, the time interval (0, 6hrs] associated with the directed edge from *Lane Reversal Order Issued* event to *Lane Reversal Activated* event indicate that lane reversal must be activated within 6 hours from the time the lane reversal order is issued. An edge in the VTS diagram can also be labeled with one or more forbidden events. For instance in Fig 4(a), the event *Road Inspection Begin* is labeled as a forbidden event between the events *Road Clearance Begin* and *Road Clearance Completed*. This means that *Road Inspection* cannot begin before completion of *Road Clearance*.

For verification of the emergency response process, we need to ensure that the process will always be executed correctly. In other words, we need to verify that all possible traces that can be composed from the individual automaton of collaborating agencies match with the VTS diagram corresponding to the process specification. Clearly a straightforward application of the VTS scenario matching approach will not work for verification of process specifications due to the existential semantics.

There are two ways by which we can verify the correctness of a given process using the scenario matching approach. One way is to exhaustively search all the traces of the automata and ensure that each trace matches with the given scenario. The second way is to exhaustively compute all scenarios that differ with the given scenario in at least one property. We refer to a scenario *S'* that differs with *S* in at least one property as a complementary scenario of *S*. The differences in properties between *S* and *S'* may include but are not limited to changes in the ordering of events, changes in the type of time interval constraints between events Fig 4(b) and Fig 4(c) show two complimentary scenarios of Fig 4(a). In the complementary scenario of Fig 4(b) the temporal gap of *at most* six hours between the occurrence of the events *Lane Reversal Order Issued* and *Lane Reversal Activated* is changed to the temporal gap of *at least* six hours. Similarly, in the complementary scenario of Fig 4(c) the event *Road Inspection Begin* occurs before the event *Road Clearance Completed*. In the original scenario of Fig 4(a) the event *Road Inspection Begin* is a forbidden event and cannot occur before *Road Clearance Completed*. Note that a given scenario may have a large number of complementary scenarios. Matching any of the complimentary scenarios with any trace of the automata is sufficient to say that the process specifications are inconsistent with the rule sets of the collaborating agencies and the execution of the process may fail because of such inconsistencies. For verification of emergency response

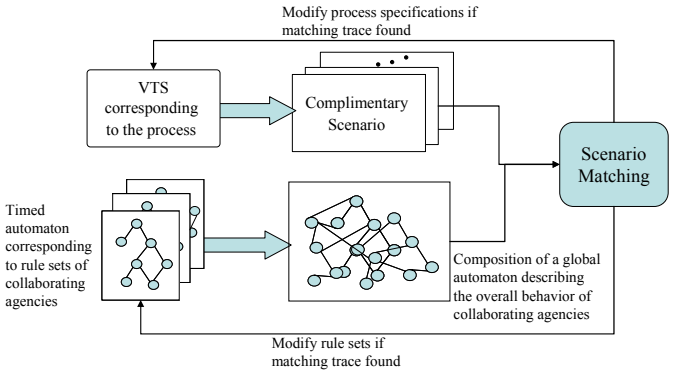

Fig. 5. Overall Verification Methodology

processes, we employ the complementary scenario matching approach discussed above. The overall verification methodology is depicted in Fig 5. The verification process requires composition of a global timed automaton from the timed automata corresponding to the rule sets of each collaborating agency. This global automaton defines the overall behavior of all collaborating agencies and can be composed by computing the product of the automaton of each agency. Once the global automaton is composed, it is analyzed for traces that match with any of the complementary scenarios of the given process. For scenario matching, we use the model checking approach described in [7]. In case a trace that matches with any of the complementary scenarios is identified, it is reported to the relevant agencies for making changes in the process design or the rule sets. The verification process is repeated whenever there is a change in the process design or rule sets.

Finding an exhaustive set of complimentary scenarios is a challenging task due to the large number of such scenarios. Currently, we do not have a formal algorithm for finding the set of complimentary scenarios and we use *ad hoc* methods for this purpose. As a future work we plan to address the issue of complimentary scenarios in the context of process verification. Specifically, we plan to develop efficient algorithms for generating the minimal set of complementary scenarios that can detect any inconsistent specification of the given process with respect to the combined rule sets of collaborating agencies.

V. IMPLEMENTATION

The initial implementation of our methodology aims at providing, through an open and interoperable system, easy-to-use knowledge representing tools for environmental risk management experts. We build on the proposed framework of our approach. The Event Analysis component is implemented through Rule Editor Client module. The basic implementation of the Verification component is achieved through the Rule Editor Server module. The Repository Servlet represents an initial implementation of the Semantic based Service Discovery component of the proposed framework. Fig 6 illustrates the architecture and implementation of our approach, which includes the rules editor server and a client side browser that is used to access the application. The views

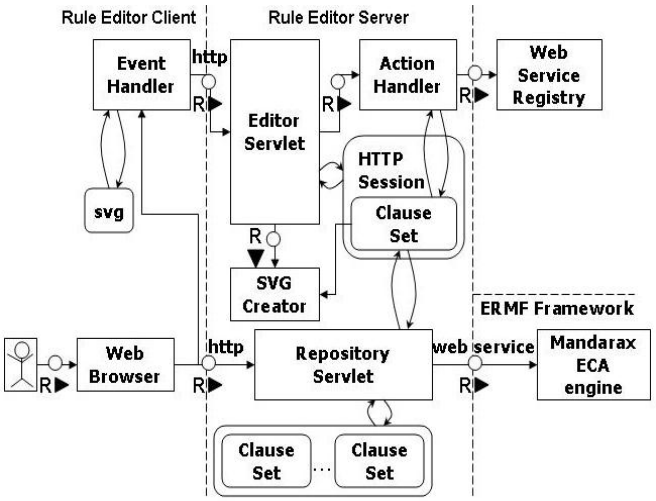

Fig. 6. Initial implementation of the proposed emergency management architecture of Fig 1

are provided by two Java servlets running on server side. Additionally, the application connects to the Web Service Registry and the runtime framework.

1) Rule Editor Server

We make use of the SAP NetWeaver 2004s J2EE based server for implementing the Rule Editor Server . The entry point, *RepositoryServlet* servlet, exposes the list of available rule sets (called clause set in the editor). The *EditorServlet* handles user modification of a rule set, and the display of available web services. Specifically, the EditorServlet reads

the current rule set from the HTTP-Session object. Next, the user-requested modification action is performed on this clause set by the *ActionHandler*. *ActionHandler:* The *ActionHandler*, called by EditorSerlvet- reads the parameters from the HTTPGET-request and determines the specific action to be performed. Once the specified action is achieved, the *SVG Creator* component converts the updated clause set into a svg representation, which is returned to the client. Eventually, the modified clause set is stored in the HTTP-Session for the next request. For Web Services display, EditorServlet requests the list of services from the Web Service registry (in our case the NetWeaver UDDI server) and transforms this list into an XML file; which is then returned to the client.

2) Rule Editor Client

The Rule Editor Client consists of a html page (editor.html) containing an svg document (*editor.svg*). This svg is provided with JavaScript that issued for handling actions done in the editor. We developed two script files, namely *svgdiagram_transforms.js* and *editor.js*. The editor.js handles all the user actions such as drag and drop of all the elements, changing the properties in the properties area, marking an object, retrieving data from the servlet, or handling the events from the "button bar" on top of the editor. This script also contains methods for inserting the data received from the server in the svg-dom (*updateSvg()*); filling the properties area with parameters of the currently selected object (*doMarkObject()*). editor.js also contains the different selectionLists, textBoxes and buttons together with their event handling methods.

3) The Framework Connection

Framework refers to the runtime system that evaluates the currently activated rules sets. The rule set is stored as a Mandarax ZKB-file and the url of this file is sent immediately to the Mandarax ECA Engine [11]. This engine then downloads and opens the file. The connection to the service registry is done using the [26] uddi4j open source project. The class org.uddi4j.client.UDDIProxy provides the access to the server. Here, the UDDI service of the Netweaver is used. Within the class *com.sap.crnce.ermf.wsclient.DynamicWSClient*, two methods have access to the UDDI server; *getWSDLByName*() and *getServices*(). The first one returns the url where the wsdl of the given service can be found. The service is described by its name. The second method reads the full list of services from the specified business entity.

VI. Conclusion and Future Work

In this paper, we present a service-oriented based approach for emergency management and response. The proposed approach addresses three major challenges: identification of potential threats and disambiguation of alarms; discovery and selection of relevant web services; and verification of emergency response process specifications. We employ multi-stream data mining for event prediction and alarm disambiguation. For Web service discovery and selection, we present a multi-step methodology consisting of ontology guided service characterization, service parameter association mining, and service request expansion. For verification of emergency response process we use a timed automata based approach that determines the consistency of the process specifications with respect to the rules/policies of collaborating agencies.

In our future work we plan to address the issue of complementary scenarios in the context of process verification. We plan to develop efficient algorithms for generating the minimal set of complementary scenarios that can detect any inconsistent specification of the given process with respect to the combined rule sets of collaborating agencies.

References

[1] N. R. Adam, V. P. Janeja, and V. Atluri, "Neighborhood Based Detection of Anomalies in High Dimensional Spatio-temporal Sensor Datasets," Proc. of the ACM Symposium on Applied Computing (ACM SAC), March 2004, Cyprus

[2] N. Adam, V. Atluri, V. P. Janeja, J. Vaidya, M. Youssef., A. Snbl, C. Bornhoevd, S. Raiyani, T. Lin, J. Cooper, J. Paczkowski "Semantic Graph based Knowledge Discovery from Heterogeneous Information Sources", Working Together: Conference on Public/Private R&D Partnerships in Homeland Security, April 2005, Boston, MA.

[3] N. Adam, V. Atluri, V. P. Janeja, A. Paliwal, M. Youssef, S. Chun, J. Cooper, J. Paczkowski, C. Bornhoevd, I. Nassi, J. Schaper, Semantics-based Threat Structure Mining, the Seventh Annual International Conference on Digital Government Research (dg.o), May, 2006, San Diego, CA.

[4] N. Adam, A. Kozanoglu, A. Paliwal, and B. Shafiq, "Secure Information Sharing in a Virtual Multi-Agency Team Environment," Proc. of the 2nd International Workshop on Security and Trust Management, in conjunction with ESORICS-06, 2006.

[5] M. Ankerst, M. M. Bruenig, HP. Kriegel, J. Sander. OPTICS: Ordering Points to identify the clustering structure, Institute of Computer Science, University of Munich. ACM SIGMOD '99

[6] V. Braberman, D. Garbervetsky, A. Olivero, "Improving the Verification of Timed Systems Using Influence Information," Proc. of the 8th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, Lecture Notes In Computer Science, Vol. 2280, 2002, pp. 21 – 36.

[7] V. Braberman, N. Kicilof, and A. Olivero, "A Scenario-Matching Approach to the Description and Model Checking of Real-Time Properties," IEEE Transactions on Software Engineering, Vol. 31, No. 12, pp. 1028-1041.

[8] B Berthomieu and M Diaz, "Modeling and Verification of Time Dependent Systems using Time Petri Nets," IEEE Transactions on Software Engineering, Vol. 17, No. 3, March 1991, pp. 259-273.

[9] D. Bedrardi, D. Calvanese, G. D. Giacomo, M. Lenzerini, and M. Mecella, "Automatic Composition of E-Services that Export Their Behavior," Proc. of International Symposium on Service Oriented Computing, 2003, pp. 43-58.

[10] A. B-Can, T. Bultan, and X. Fu, "Design for Verification for Asynchronously Communicating Web Services," Proc. of the Fourteenth International World Wide Web Conference (WWW 2005), pp. 750-759.

[11] J. Dietrich, Alexander Kozlenkov, Michael Schroeder, Gerd Wagner: Rule-Based Agents for the Semantic Web, Journal on Electronic Commerce Research Applications, 2003.

[12] S. T. Dumais, “LSI meets TREC: A status report." In: D. Harman (Ed.), The First Text REtrieval Conference (TREC1), National Institute of Standards and Technology Special Publication 500-207 , pp. 137-152

[13] M. Ester, H. -P. Kriegel, J. Sander, and X. Xu. "A density-based algorithm for discovering clusters in large spatial databases.” KDD'96.

[14] X. Fu, T. Bultan and J. Su. "Conversation Protocols: A Formalism for Specification and Verification of Reactive Electronic Services." Theoretical Computer Science, Vol. 328, No. 1-2, November 2004, pp. 19-37.

[15] V. Ganti, J. Gehrke, and R. Ramakrishnan, “Mining Data Streams under Block Evolution,” SIGKDD Explorations, Vol. 3, No. 2, 2002, pp. 1-10.

[16] J. Garofalakis, Y. Panagis, E. Sakkopoulos, A. Tsakalidis, “Web Service Discovery Mechanisms: Looking for a Needle in a Haystack?”, International Workshop on Web Engineering, 2004

[17] F. Jahanian and A. Mok, “A Graph-Theoretic Approach for Timing Analysis and its Implementation,” IEEE Transactions on Computers, Vol. 36, No. 8, pp. 961-975.

[18] V. P. Janeja, V. Atluri, A. Gomaa, N. R. Adam, C. Bornhövd, and T. Lin, “ DM-AMS: Employing Data Mining Techniques for Alert Management,” NSF National Conference on Digital Government, 2005, pp.103-111

[19] V. P. Janeja, V. Atluri, J.S.Vaidya, and N. R. Adam, “Collusion Set Detection,” ISI 2005.

[20] D.G. Kleinbaum, L.L. Kupper, and K.E. Muller. Applied Regression Analysis and Other Multivariable Methods. PWS Publishing Co. Boston, MA, USA. 1988.

[21] M. Last, Y. Klein, and A. Kandel, “Knowledge Discovery in Time series Databases,” IEEE Transactions on Systems, Man, and Cybernatics, Vol. 31, No. 1, February 2001, pp. 160-169.

[22] http://www.ncdot.org/traffictravel/emergencyinfo/

[23] J. Ostroff. Temporal Logic of Real-time Systems. Research Studies Press, 1990.

[24] A.V. Paliwal, . Adam, N., and C. Bornhövd, RUTGERS University - CIMIC Technical Report, "Web Service Discovery: Adding Semantics through Service Request Expansion and Latent Semantic Indexing".

[25] S. Papadimitrios, J. Sun, and C. Faloutsos, “Streaming Pattern Discovery in Multiple Time-Series,” Proc. of the 31st VLDB Conference, Trondheim, Norway, 2005, pp. 697-708.

[26] http://www.oasis-open.org/committees/uddi-spec/doc/tns.htm

[27] H. Xiong, P. Tan, & V. Kumar, “Mining Strong Affinity Association Patterns in Data Sets with Skewed Support Distribution”, IEEE International Conference on Data Mining (ICDM), 387-394.

[28] B.-K. Yi, N. Sidiropoulos, T. Johnson, H. V. Jagadish, C. Faloutsos, and A. Biliris, “Online Data Mining for Co-evolving Time sequences,” Proc. of the 16th International Conference on Data Engineering, 2000, pp. 13-22