# DM-AMS: Employing Data Mining Techniques for Alert Management

Vandana P. Janeja[1], Vijayalakshmi Atluri[1], Ahmed Gomaa[1], Nabil Adam[1], Christof Bornhoevd[2] and Tao Lin[2]
[1]CIMIC and MSIS Department, Rutgers University
[2]SAP Research Labs, Palo Alto, CA
{vandana,atluri,ahgomaa,adam}@cimic.rutgers.edu
christof.bornhoevd@sap.com

## ABSTRACT

Alert management plays a critical role in many application domains including *homeland security* and *natural disaster management*, to allow timely and well-informed decisions. The major challenge faced by these systems is that the number of incoming alarms is overwhelming and some of the alarms are false positives. In this paper, we present an *alert management system* (AMS) that generates meaningful alerts from alarms received from different sensors. The alert generation module of our system (i) flags and eliminates potential false positives by characterizing the region into uniformly behaving neighborhoods, (ii) generates *aggregated alerts* from the alarms by employing density based clustering techniques and identifying the overlap among clusters, and (iii) identifies the *dynamic flow* of the alerts by integrating scientific models that characterize the behavior of sensor parameters. Once the alerts are generated our *customized dissemination module* disperses the alerts on the need-to-know basis to the individuals and agencies involved. This module adheres to the National Incident Management System (NIMS) and the National Response plan (NRP) protocols. To implement these protocols, we utilize the Common Alerting Protocol (CAP), which is an XML nonproprietary data interchange format. Finally, our *GIS module* displays the alerts through a user-friendly interface.

## 1. INTRODUCTION

Alert management plays a critical role in many application domains including *homeland security* and *natural disaster management*, to allow timely and well-informed decisions. Currently alert management systems (e.g. [G04, FP97, FP99, PK02]) are based on scoring of alarms and creating profiles for identifying true alert vs. a false positive. However, in many cases training data may not be available for identifying scores or profiles. Since the response time is a critical factor in these applications, a training based system is not a suitable solution.

The series of alarms received in a complex application domain is composed of the raw messages received from multiple sources and an alert is the processed and aggregated output generated from these alarms. Typically the number of incoming alarms is so overwhelming, it is essential to eliminate the false positives from genuine alarms, and to aggregate the alarms with specific characteristics to generate an alert. This way, it reduces the clutter and help decision makers understand the situation better so that they can make well informed decisions. Furthermore, a decision maker would want to analyze the effect of this alert by combining it with the situation at hand. For example, the geographic region affected of an alert generated from a radiological sensor may be better judged by integrating this information with plume model. Our proposed *alert management system* (AMS) for reducing false positives and generating *aggregated alerts* from a series of alarms is motivated by the concept of knowledge reformulation [M97, L00].

After generating the alerts, it is necessary to send only the need-to-know information to the concerned individuals at different agencies involved in the decision making process. Our AMS includes a *customized dissemination module* to disperse the alerts on the need-to-know basis, and a *GIS module* to display the alerts through a user-friendly interface. In essence, our AMS presents an approach to develop the tools required to *aggregate* and *disseminate* these alarms.

In the following, we present a motivating scenario based on the working of an Emergency operations center (EOC).

***Motivating example***: *In the event of an emergency (natural or man-made) the chaos at the epicenter of a disaster calls for immediate and accurate communications between first responders and emergency management organizations. Regional organizations are beginning to fill this niche by promoting efforts to coordinate and manage information sharing (such as [L05]). However, while strides have been made in this direction, coordination between various agencies involved is lacking [J04]. Typically an emergency scenario requires the management of multiple resources namely important infrastructure such as bridges, tunnels, bus terminals, airports, and the data feeds from multiple sources such as radiological, chemical and biological sensors, as well as video surveillance cameras that may feed the video feed in real time, and audio reporting devices. In the event of an emergency the EOC would receive alarms sent by various entities in this infrastructure such as officers in the field, alarms generated from a number of standalone software components residing at different agencies (such as the Police, Fire, Health etc.), alarms generated by numerous sensors and so on. The number of alarms thus becomes overwhelming and difficult to interpret at the time of an emergency where quick reaction is required based on the data feed. Therefore it becomes important to integrate these numerous alarms into a smaller coherent set of alerts. Moreover, once these alerts are generated, they should be appropriately directed to the right people at the right time. Thus an alert specifically related to the fire department for a specific location should not be sent to the fire department another location unless there is a need for specific collaboration between the agencies and only if this information sharing is feasible under the*

*policies of each of the departments. This would prevent the field officers from being overwhelmed by information.*

As can be seen from the above scenario, the following three different requirements emerge: 1) there is an overwhelming number of alarms generated by different agencies, people and sensors which often cause confusion, and would be difficult to interpret when considered in this unfiltered, non-aggregated form. 2) There could be many false positives in this stream of alarms and it is critical for high level decision making to determine the true alerts vs. false positives, and 3) For the decision makers it may be required to associate an alert with a dynamic flow of the alert over a period of time, thus there is a need to integrate the alerts with certain models to interpret and project the alert scenario in a dynamic setting to help in making more informed decisions.

To meet the above requirements, we undertake the following three steps for the generation of aggregated alerts. 1) Eliminate ***false positives*** from the stream of alarms, as much as possible, for this we focus on alarms generated primarily from sensors 2) Generate ***aggregated alerts*** from mutually related alarms for better interpretation. This aggregation is based on the mutual relationship of the attributes qualifying the alarms (such as location, originating agency, content etc.) and 3) Generate a ***dynamic flow alert*** to integrate the alarms with certain external models to generate alerts for high level decision making, for example plume models, traffic-flow models, etc.. The primary difference between the aggregated alert and a dynamic flow alert is that, the former is based on a past event, whereas the latter is based on a future event. The current system modules have been implemented however the evaluation in a real emergency management setting has not been carried out due to various constraints.

The rest of the paper is organized as follows. In section 2 we describe the alert generation process. In section 3 we describe the alert management system, in section 4 we describe the related work and finally in section 5 we discuss the conclusions and future work.

# 2. ALERT GENERATION

A series of alarms may be generated from a complex system consisting of softwares, agencies, sensors etc. For the purpose of this paper we consider all these sources as sensors generating alarms, however this approach can be generalized to other types of sources. The Alert generation for such a complex system would involve the following steps 1) Alarm preprocessing: for the identification of false positives in the streams of alarms. 2) Aggregated Alert Generation: The series of alarms are interpreted based on different relationships between the alarms, which will lead to an aggregation of alarms into alerts. Aggregated alerts are those alerts, which are identified based on clustering of alarms. If a cluster of alarms has a significant number of alarms from a single or multiple sources then this cluster may be termed an aggregated alarm, secondly if there is a set of clusters in which no one cluster has a significant number of alarms then we look for a mutual overlap of the clusters and this overlap becomes the aggregated alert. 3) Dynamic flow alert generation: The alert generated will be integrated with a relevant model for generating the dynamic flow of the alert based on some factors determined by the external model. Here we identify based on the input of a model the dynamic flow of an alert starting from a source and flowing in a certain direction given the time lag and direction of the flow of the alert. The alert generation process is shown in figure 1.

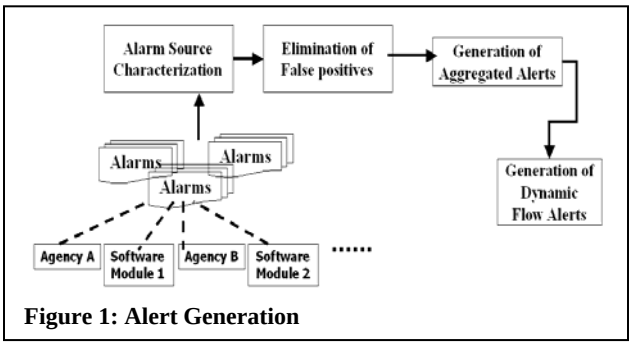


**Figure 1: Alert Generation**

We next describe the alert generation in detail. We first begin with an understanding of alarms.

## 2.1 Alarms

Various entities involved in a system, such as agencies, different software components, sensors etc., may generate alarms at any given time. For this paper we primarily consider alarms generated from different sensors. We first explain the characteristic of an alarm.

Each alarm $a_i \in A$ is associated with (i) a set of attributes $t= \{t_{i1}, t_{i2}, \ldots, t_{im}\}$, and (ii) a source $s_i$ such that each source itself may have some associated features $f = \{f_1, f_2, \ldots, f_p\}$ and location $(s_{ix}, s_{iy})$ denoting its (x, y) coordinates.

The attributes associated with alarms may include originating agency, timestamp of the alarm, and content of the alarm itself such as information about some chemical detected by the sensor or some information typed by a person. The source attribute of the alarm essentially points to the coordinates for the origin of the alarm. The source of the alarm is further qualified by features for instance in case of a sensor the features may be the presence of a chemical factory, a railway line, average temperature at the sensor, agricultural production, the area covered by water, number of importers located in the area, and so on. The reason these features are an important part of the analysis is because these features determine the behavior of the sensors [AJA04]. Similarly, these features may decide whether a sensor may behave similar to other sensors in the region. We will utilize the features associated with the source of the alarm and the attributes of the alarm itself to determine alerts and their validity in an emergency scenario.

## 2.2 Alarm Source Characterization

The alarms could be generated from a single source as a series of bursts, or from multiple sources. To distinguish whether these alarms are true alerts or due to a malfunction of a sensor, we examine if the sensors in the *similarly behaving region* also are generating any alarms. This helps us to reduce the false positives. In the following, we first characterize similarly behaving regions (or similarly behaving sensors), using the concept of *neighborhood*.

We generate a *micro neighborhood* [AJA04] around each source based on the concept of Voronoi polygons. Using the similarity among the attributes represented by semantic and spatial relationships among them, we merge the micro neighborhoods to form a larger region, called the *macro neighborhood*. We employ

the Jaccard coefficient to quantify the similarity. The goal of constructing a macro neighborhood is to characterize a set of sources. We briefly describe this characterization process in the following steps:

*1. Creation of micro neighborhoods:*

For this we utilize the information provided with the source of the alarms namely the coordinates and the features of the source of an alarm. The features f = {$f_1$, $f_2$, …,$f_p$}of each source $s_i$ are transformed into categorical values (binary) to form a feature vector $f'_i$ = { $f'_1$, $f'_2$, …, $f'_p$}. For example, if a railway line is present at the sensor location, its corresponding feature value is set to 1, otherwise it is set to 0.

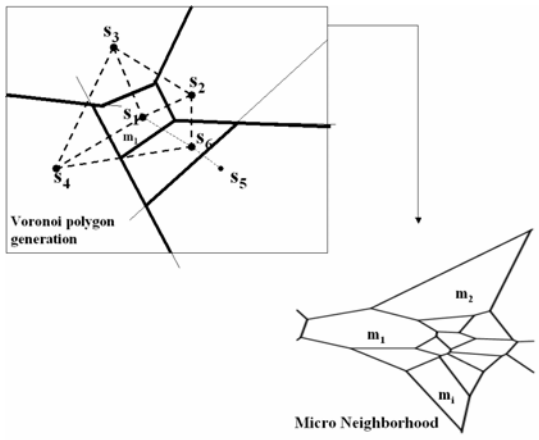


**Figure 2 : Formation of Micro Neighborhood**

The micro neighborhood is constructed based on the concept of Voronoi tessellations [OBS00], where each source is allotted its region of influence in terms of an intersecting half plane. The formation of a micro neighborhood is illustrated in figure 2 adapted from [JAA04]. To construct a Voronoi diagram, first two sources are connected by a line segment, which is then bisected into two half planes. The intersection of a series of such bisecting lines generates a polygon around each source. As sources are added, more half planes are formed, and the region of influence of the source is formed by the intersection of the half planes. Each Voronoi polygon is comprised of the entire proximity information about a source in an explicit and computationally useful manner. We adapt the definition of micro neighborhood and region of dominance as follows [JAA04].

**Definition 1 [Micro Neighborhood]:** Given a set of sources s={$s_1$, $s_2$, …,$s_n$}, a micro neighborhood $\mathbf{m_i}$ is the polygon bounded by a Voronoi polygon V($s_i$).

The micro neighborhood essentially represents the dominance of a source over another source [OBS00]. Thus, a feature located on one side of the bisector is closer to that half plane than the other.

**Definition 2 [Region of Dominance]:** Given two sources $s_i$ and $s_j$, and their feature vectors $f'_i$ and $f'_j$ such that $f'_{ik} \in f'_i$ and $f'_{jk} \in f'_j$, we define the *dominance* of $s_i$ over $s_j$, as follows: dominance($s_i$, $s_j$) iff $d(f'_{ik}, s_{ik}) \leq d(f'_{jk}, s_{jk})$}, where d is a distance function.

Thus, a micro neighborhood is a bounded polygon $m_i$ around each source $s_i$, encompassing this knowledge of dominance for each source.

*2. Creation of Macro neighborhoods:*

Given the micro neighborhoods generated above, we next describe the creation of macro neighborhoods. Given that two micro neighborhoods have a spatial relationship, namely topological (adjacent, inside, disjoint, ..), direction (north, south, east, west, north-east …), or distance relationship [EKS97]. For our purposes we focus on identifying the topological relationship of adjacency between two sources. We next identify a semantic relationship between the two. The semantic relationship is based on computing a similarity between the feature vectors of the two sources. This similarity can be computed using the Jaccard coefficient [KR90]. Semantic relationship is defined as follows.

**Definition 3 [Semantic Relationship]:** Given two micro neighborhoods $m_i$ and $m_j$, and the corresponding feature vectors $f'_i$ and $f'_j$, the degree of the semantic relationship between $m_i$ and $m_j$, denoted by sm($m_i$, $m_j$ ) = JC($f_i$,$f_j$), such that JC($f_i$,$f_j$) > $\phi$ where $\phi$ is a threshold value for the level of similarity .

Based on the spatial and semantic relationship identified we create a macro neighborhood.

**Definition 4 [Macro Neighborhood]:** Given two micro-neighborhoods $m_i$, and $m_j$, we say that they are part of a macro neighborhood $M_i$ if there exists a spatial relationship ($m_i$, $m_j$) and semantic relationship ($m_i$, $m_j$) $\geq$ $\phi$, where $\phi$ is a threshold for the similarity between the feature vectors.

Thus the contour of the macro neighborhood is formed by eliminating the common edge(s) between the adjacent micro neighborhood polygons and considering only their outer edges. The formation of a macro neighborhood is illustrated in figure 3 adapted from [JAA04]. This essentially forms a baseline characterization in a normal scenario. We next outline the steps of the algorithm shown in figure 5 in section 2.6.

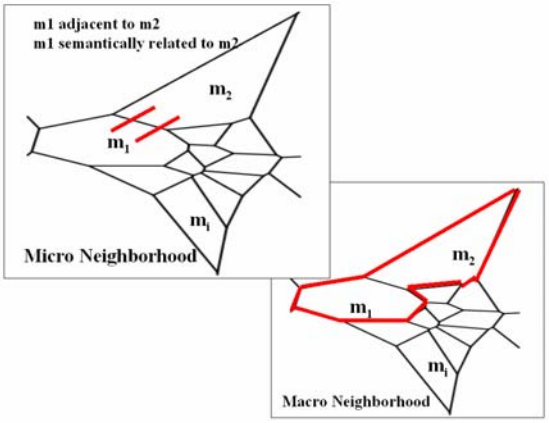


**Figure 3: Formation of Macro Neighborhoods**

**Given**

a) A set sources associated with features:

**Step 1:** Generate Micro neighborhood around each source

**Step 2:** Generate Macro Neighborhoods

2.1) Iteratively, check for spatial and semantic relationships between two micro neighborhoods $m_i$ and $m_j$

2.2) If there exists a spatial and semantic relationship then $m_i$ and is similar to $m_j$ and for Macro Neighborhood

## 2.3 Elimination of False Positives

Once the alarm occurs, the originating source can be compared to other sources in the macro neighborhood to determine if this sensor is generating false positives due to a malfunction or error, or this is truly an alert.

In order to identify the false positives, we form the alarms into clusters based on attributes associated with them. We adapt the notion of density based clustering algorithm [EKSX96] and define clusters of alarms:

**Definition 5 [Clusters of Alarms]:** Given a set of Alarms A={$a_1$, $a_2$,…,$a_n$}, where each $a_i$ is qualified by a set of attributes t= {$t_{i1}$, $t_{i2}$, …,$t_{im}$}, a cluster $c_x$={ $a_1$, ..$a_p \in$ A | for any alarm $a_q$ to any other alarm $a_r$ , ed($a_q$ , $a_r$) < d from ρ such alarms}, where ed is the euclidean distance between the two attribute vectors of the alarms, d is a distance threshold and ρ is the number of alarms to which this alarm is close in terms of distance.

Thus if an alarm is close to a certain number of points it iteratively forms a cluster until the clustering converges. Here the similarity in the attributes is captured using distance for numeric alarms.

We assume the set of clusters is C. We next formally define a false positive.

**Definition 6** [**False Positive**]**:** Given a cluster c comprising of a set of alarms A= {$a_{i1}$, $a_{i2}$, …,$a_{im}$}, such that the source of each $a_k \in$ A is $s_i$. Let $s_i \in M_p$. We say that A is a set of false positives, fp, (i) if $|A| \geq \sigma$ where σ is a significance threshold and (ii) there does not exist a cluster $c^{|}$ comprising of a set of alarms $A^{|}$ ={$a_{j1}$, $a_{j2}$, …,$a_{jo}$} such that the source of each $a_l \in$ each $a_k \in$ A is $s_j \in M_p$ and $|A^{|}| < \sigma$.

Essentially, for each cluster we identify the set of sources associated with each cluster. We next check if only one source in a given Macro neighborhood is generating alarms in which case this cluster is flagged as a possible false positive for further investigation of the source. While the algorithm is depicted in figure 5 in section 2.6, in the following, we outline the detailed steps.

**Given:**

a) A set of alarms with their attributes and sources associated with features

b) Threshold values for – the minimum number of clusters, significance threshold for number of alarms in a cluster

c) Macro Neighborhoods created in section 2.2

**Step 1 :** Generate clusters of alarms based on attributes. For each cluster identify the set of sources associated with each cluster, Cardinality of cluster which is the number of alarms in cluster

**Step 2 :** Check if only one source is generating alarms then flag for further investigation of source

2.1) If cardinality of a cluster is greater than the significance threshold

a) Check if all alarms in this cluster are from one source

a.1) Select the Macro Neighborhood associated with the source in the cluster

a.2) For every source s in the Macro Neighborhood such that this source is not equal to the source in the cluster

a.2.1) if the source s is in not in any other cluster it implies that other sources in the macro neighborhood are not generating alarms

a.2.2) Flag this cluster as possible false positives for further investigation

## 2.4 Generation of Aggregated Alerts

After removing the clusters representing the false positive, among the remaining clusters of alarms, two cases may occur: a) A cluster comprises of a significant number of alarms but these alarms do not belong to one single source in a macro neighborhood and b) No single cluster has a significant number of alarms. In the following we discuss how aggregated alerts can be generated in each of these two scenarios. In case of the first scenario, we will generate an aggregated alert if there is any cluster which has alarms greater than a *significance* threshold, whereas in case of the second scenario, we will identify the overlap between the clusters to find *mutually related alarms* to generate an aggregated alert. It is clear that higher the overlap more strongly related are the alarms.

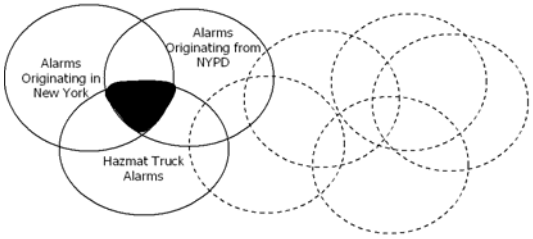


**Figure 4: Mutually related alarms**

We utilize *entropy overlap* [BEX02] for identifying the overlap of the clusters. Entropy essentially measures the quality of the clustering in terms of how the elements in the cluster have been allocated. We next define entropy overlap.

**Definition 7 [Entropy Overlap]:** Entropy overlap for a cluster $c_i$, is denoted by EO($c_i$) and is defined as the distribution of the alarms of a cluster $c_i$ over all the remaining clusters.

$$EO(c_i) = \sum_{a_j \in c_i} [-1/b_j \cdot \ln(1/b_j)]$$

Here b is the number of attribute in the alarm. The attributes may be the originating agency, temperature values, HAZMAT description and so on. The entropy becomes 0 if all alarms $a_j$ of $C_i$ contain one frequent attribute and as frequent attributes increase the entropy increases. The lower the entropy the purer is the structure of the clusters. Thus the aim of the clustering would be to minimize the entropy and on the other hand to identify the overlap one would need to identify the clusters with the highest entropy. Essentially, this will allow eliminating the alarms that are not mutually related, thereby leading to the filtering of the alarms.

Based on the above we define an aggregated alert.

**Definition 8 [Aggregated alert] :** Given a set of clusters of alarms C= {$c_1$, $c_2$, …,$c_e$}, an aggregated alert g is the cluster of alarms $c_f$ if $|c_f| \geq \sigma$ such that $c_f$ is distributed over sources $s_j \in M_p$ , where $M_p$ is a macro neighborhood, if $|c_f| < \sigma$ an aggregated alert g is the set of alarms in EO($c_f$).

Once an alert is generated we would also like to associate sources with it. The set of sources of the alarms is the sources of the alarms in the cluster or in the other case the sources of the alarms in the overlap. These sources can be aggregated by aggregating the feature vectors of all the sources thus generating a composite feature vector of the source of the alert. We defer the aggregation of sources for future research as it may involve aggregation at a policy level as well in case of the sources being agencies or other government or private entities.

We next outline the steps in the algorithm for generating aggregated alerts, the algorithm is depicted in figure 5 in section 2.6.

**Given:**

a) Macro Neighborhoods created in section 2.2

b) Clusters generated in section 2.3

c) Significance threshold for number of alarms in a cluster, entropy threshold

**Step 1:** If cardinality of a cluster is greater than significance threshold

1.1) Check if this cluster is not flagged as false positive

flag this cluster and raise ALERT

**Step 2:** If cardinality of a cluster is < significance threshold

2.1) Iteratively compute Entropy Overlap

2.2) If the entropy is greater than the entropy threshold

flag cluster overlap and raise ALERT

## 2.5 Generation of Dynamic Flow Alerts

Once we determine that we have a true alert, it may be useful to determine a dynamic movement of the alert. Thus we not only know that there is an alert but also identify the dynamic flow of this alert. The dynamic flow can be based on several factors based on the type of the alert. For instance if the alert is related to a chemical plume then the dynamic flow will be determined by the environmental factors affecting the flow of the plume, where as in the case of water contamination alert the factors would be based on the dynamics of the flow of the water. We consider the dynamic flow by including time as a factor and generate the dynamic flow alert by integrating an alert with the model.

Essentially a model T provides parameters as the set of sources with the associated time lag values δ between every pair of sources in the movement of an alert and direction of flow starting from source s. These parameters provide an explanation of the dynamic nature of the environment. For instance the model may provide two alerts $A_1$ corresponding to a sources $s_1$ and $A_2$ corresponding to a sources $s_2$. Assume the model provides the time lag between $s_1$ and $s_2$ as $\delta_{12}$. Based on the parameters from the model we next define the dynamic flow alert. The value δ may be a factor determined by a model for a certain type of alert. Thus if there is a plume flow in a region, the plume model would determine the time lag which can then be associated with the time of the alert taking place and this can be seen as a flow in a characterized region. In some cases the model may only reflect the effect of scientific parameters, however there may be other factors influencing the flow of the alert such as political, demographic, agricultural, economic factors etc. Thus in order to get a dynamic flow alert which reflects all such factors we also consider the macro neighborhood which encompasses lot of these features and can be easily adapted for multiple types of factors. We next formally define a Dynamic flow alert.

**Definition 9 [Dynamic Flow Alert]:** Let T be a model comprising of a parameter of a set of sources $\{s_1, s_2, .. s_n\}$ and the time lag $\delta_{ij}$ between every pair of sources $s_i$ and $s_j$. Let $s_k$ be the source of an aggregated alert. Every source $s_l$ such that $\delta_{kl} > 0$ is a dynamic flow alert DA, such that $s_k$ and $s_l \in M_q$ where $M_q$ is a macro neighborhood

**Algorithm 1** ALERT GENERATION

**Require:** Set of alarms $A=a_1,\ldots,a_n$ where each alarm $a_i$ has attributes $t_i = t_{i1},\ldots,t_{im}$
**Require:** $s_i$ the source of each alarm $a_i$ where $s_i$ has attributes $f_i=f_{i1},\ldots,f_{ix}$
**Require:** $C_{num}$ threshold for minimum number of clusters
**Require:** $\sigma$ significance threshold for number of alarms in a cluster
**Require:** $\alpha$ Entropy threshold
**Require:** Set of model parameters time lag $\delta=\delta_1,\ldots,\delta_p$
*Function* **genAlerts**
{STEP 1A:preprocessing - identifying the baseline characterization}
**for** Every source $s_i$ **do**
{Generate $Microneighborhood$ around each source $s_i$}
**end for**
**for** Every $Microneighborhood$ $m_i$ **do**
**for** Every $Microneighborhood$ $m_j$ such that $i \neq j$ **do**
**if** $m_i$ spatially related to $m_j$ and $m_i$ semantically related to $m_j$ **then**
{$m_i$ is similar to $m_j$}
{$m_i$ and $m_j$ characterized into $MacroNeighborhood$ $M_i$}
**end if**
**end for**
**end for**
{STEP 1B: Identifying false positives}
{Generate clusters of alarms based on attributes}
{set of clusters $C=c_1,\ldots,c_l$ where each cluster $c_u$ has set of sources associated with each cluster $s_u = s_{u1},\ldots,s_{uj}$ where $cardinality_u = |alarms| \in c$}
{To check if only one source is generating alarm bursts then flag for further investigation of source}
**for** every cluster $c$ **do**
**if** $cardinality_c > \sigma$ **then**
**if** $cardinality_{s\_c} = 1$ **then**
**for** every source $z$ in $MacroNeighborhood$ such that $z \neq s_c$ **do**
**if** $z \notin s_i$ **then**
{where $s_i$ is the set of sources for all alarms }
{Flag cluster $i$ as possible false positive}
**else**
{STEP 2A: Generate $ALERT$}
{flag cluster and raise $ALERT$}
{as there are other sources in the macro neighborhood which are generating alarms then this big cluster becomes more significant}
**end if**
**end for**
**end if**
**else**
{STEP 2B: Generate $ALERT$}
**for** every cluster $d$ such that $c \neq d$ **do**
{Compute entropy $(c,d)$}
**if** $entropy > \alpha$ **then**
{flag cluster overlap and raise $ALERT$}
**end if**
**end for**
**end if**
**end for**
{STEP 3: Generate dynamic flow for the $ALERT$}
{Given from the model $\delta$ and direction of flow starting from source $s_\delta$}
{Identify $ALERT$ at source $s_\delta$}
{Identify set of sources $s_f$ in flow direction}
**for** every source $s_m$ in MacroNeighborhood of $s_\delta$ such that $s_m \in s_f$ and $\delta$ at $s_m$ is $\delta_m$ **do**
$\{ALERT_{TIME} = ALERT_{TIME} + \delta_m\}$
**end for**

**Figure 5: Alert Generation algorithm**

**Given**

a) Macro Neighborhoods created in section 2.2

b) A set of aggregated alerts with sources associated with features generated in section 2.4

c) Input from an external model the time lag values associated with a set of sources in a Micro Neighborhood, direction of flow

**Step 1:** Identify ALERT at source $s_k$ and Macro neighborhood of the source $M_k$

**Step 2:** Identify set of sources $S_f$ in flow direction present in $M_k$

2.1) **Identify for** every source in $s_l$ in $S_f$ ,such that $\delta_{kl} > 0$ append $s_l$ to DA

**Step 3:** Flag the sources and raise dynamic alert DA

We outline the algorithm in figure 5 for alert generation based on the above alert generation process.

## 3. DM-AMS SYSTEM

The Alert management system consists of two components (i) the geospatial interface and (ii) the customization module. In the following, we describe them in more detail.

### 3.1 GIS Interface

The alerts generated from the series of alarms are visualized on a GIS interface and the relevant information for the alert is displayed with it. It has been implemented using ARC IMS and ARC VIEW [ARC04] and Java. The prototype is available through a web interface. Screen shots from the prototype system can be seen in figures 6. The prototype consists of several GIS layers, which can be activated using the panel to the upper right corner of the screen. These layers include locations of different types of sensors, agency jurisdictions, locations of strategic resources such as fire hydrants, hospitals etc. Multiple layers can be combined to provide an overlapping view of strategic resources. Each layer is associated with geographic information in the form of latitude and longitude as well as additional information such as the monitoring parameters captured by the sensor, feature information of the sensors, and the like. The default initial active layer is the region. Layers generated from various stages in neighborhood formation at various levels of semantic relationships qualified by similarity across micro neighborhoods can be included.

Each layer can be queried, using inbuilt geospatial tools, for more information about each source. Thus, for a malfunctioning sensor on clicking the query option one may be able to see the sensor readings, location coordinates, feature information etc. A script is run in the background, which initiates the identification of the alert. This script sends an input to this interface to activate the alert layer. In the figure 6 the red dot indicates the alert generated by a sensor. Further, this can be modified to show the progression of the alert based on the dynamic flow alert. The interface also gives access to additional information about the alert including the alert source, the sources of alarms to generate the aggregated alert, and other relevant information.

### 3.2 Customization module

This module is responsible for disseminating the alerts to the right individuals and agencies at the right time in the right format. Essentially, this is to ensure not to flood all the agencies connected to the AMS with all the generated alerts. For example the information required by the police department is different than the information required by the fire department. Moreover, when disseminating the alerts, it adheres to the inter-organizational sharing policies, the credentials possessed by individuals, the role hierarchy relationships within an agency. For example, the information required by the chief of the fire department is

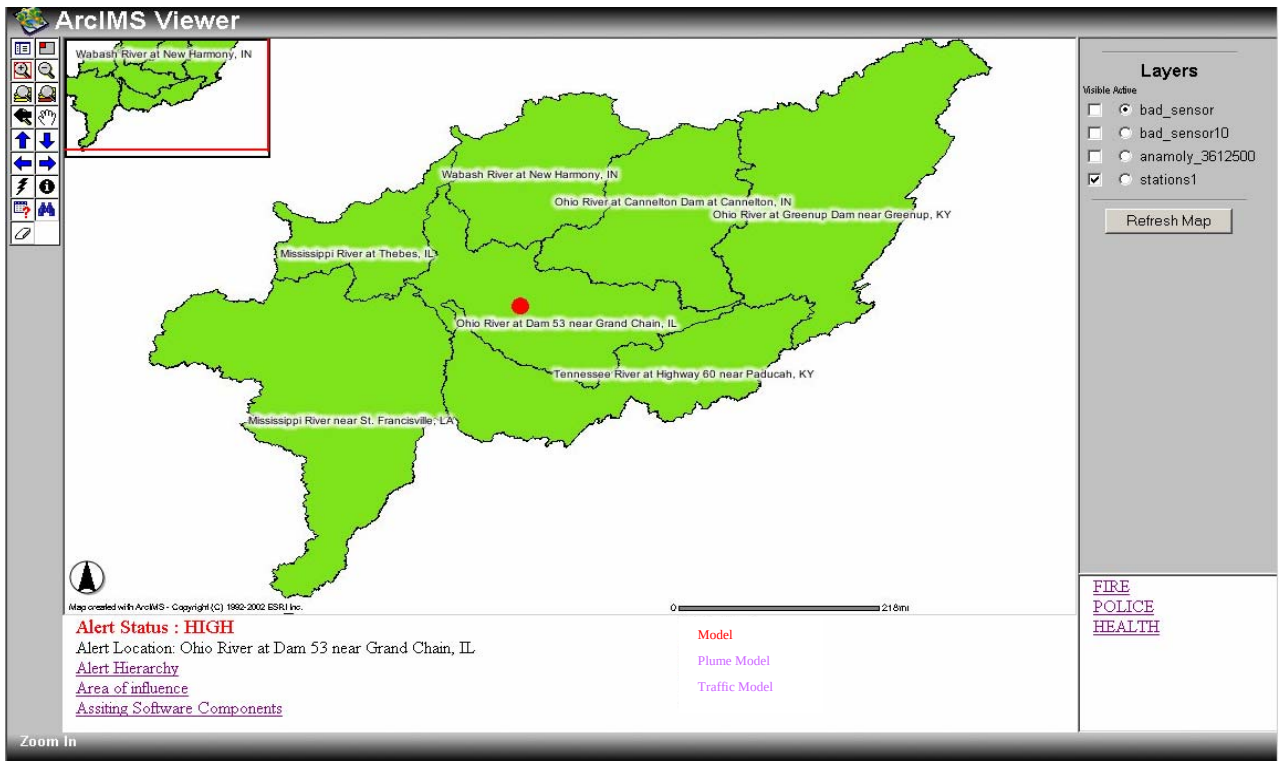


**Figure 6: An integrated GIS interface for Alert generation**

different from the information needed by the fireman. Furthermore, the content and mode of the information need to be customized based the recipient's device. For example, the alert information delivered would be different for a handheld device versus a workstation. To accomplish this, we have adapted the automatic manifestation approach developed in [AAAB01]. Therefore, the alert created on-the-fly is based on the role of the recipient and his agency as can be described in the generic publish/subscribe systems, as well as based on the recipient's device capabilities and profile .

For the underlying protocol in the customization module, we adhere to the National Incident Management System (NIMS) and the National Response plan (NRP) protocols. NIMS has been developed and administered by the Department of Homeland security to provide a consistent nationwide template to enable all government, private sector, and nongovernmental organizations to work together during domestic incidents. The National Response Plan (NRP) focuses on prevention, preparedness, response and recovery within the life cycle of an incident by establishing incident monitoring and reporting protocols. One way to allow different agency's information systems to communicate is the

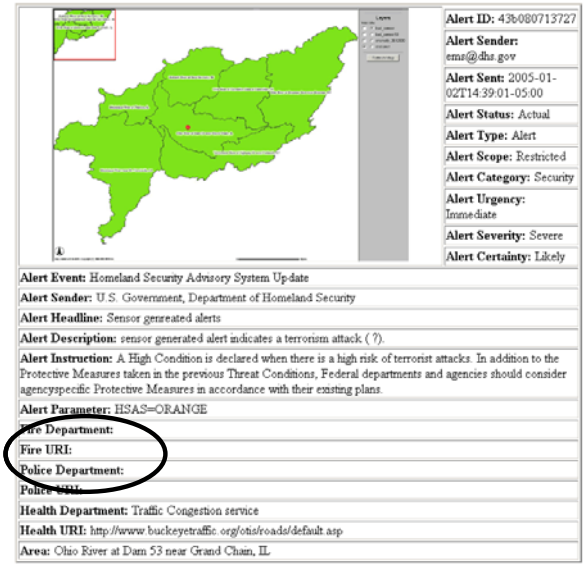


**Figure 7: Customized view for specific agency**

Common Alerting Protocol (CAP). CAP is an XML nonproprietary data interchange format that can simultaneously transmit emergency alerts through different communication networks. The Organization for the Advancement of Structured Information Standards, an international standards body, has adopted CAP as a standard.

Once the policy for manifesting the alerts is determined, the alert format is presented using CAP that provides digital message format for all types of alerts and notifications [CAP03].

CAP defines the alert message structure which includes four main segments, 1) <alert> segment, which provides the message identifier, purpose, source and status. It may contain one or more 2) <info> segments, which describe the urgency, severity, certainty, actions to be taken and related parameters of an anticipated or actual event. Each <info> segment may include one or more resource segments 3) <resource> segment provides reference to additional information such as video, image, text or

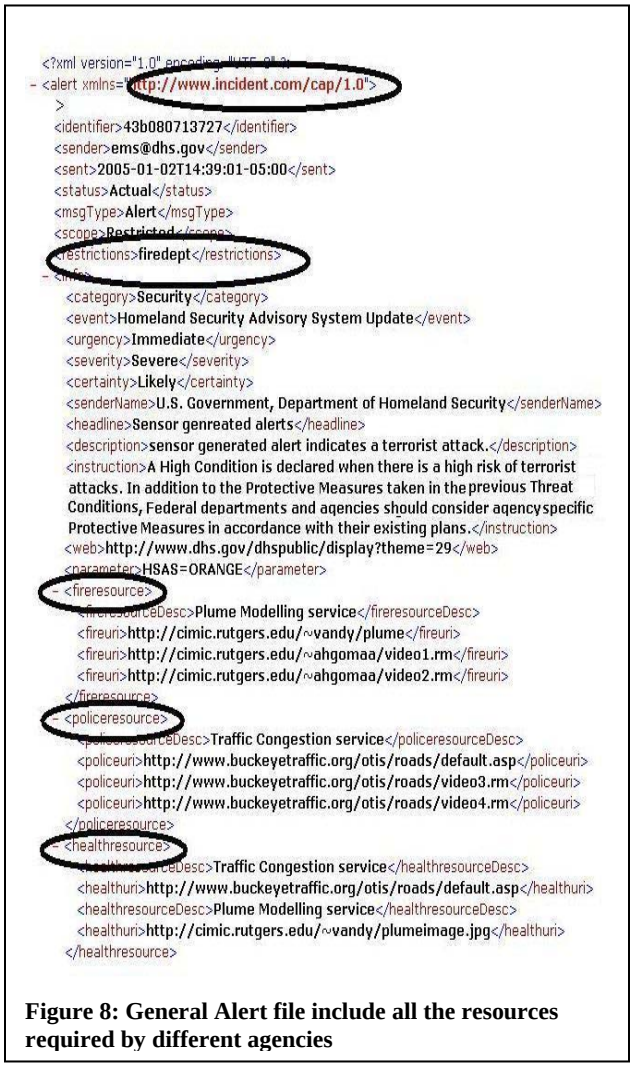


**Figure 8: General Alert file include all the resources required by different agencies**

audio file and one or more area segments 4) <area> segment describes one or more geographic areas related to the <info> segment.

The dissemination of the alert message is based on the jurisdiction policies, agency policies and the role policies. Based on the NIMS and NRP, certain protocols need to be followed in an emergency scenario. A jurisdiction policy determines which agencies should coordinate the emergency management based on the alert magnitude and area. An agency policy determines the access to certain components based on the access rights of the agency. It is used to identify which information resources should be accessed from an alert message by which individual. A role policy determines which resources to be accessed based on the role within an agency. These policies are specified using XACML, a security language standard that specifies access control over XML files [XACML01]. We do not create separate style sheets to present the XML information for each role in each agency based on each device, instead the layout of the alert is generated automatically to best convey the alert message to its diverse recipients. We have generated XSL style sheets [XSL03] based on the approaches presented in [AAGA03,GAA05]. XACML processor binds the policy with the alert and generates a new alert based on the initiator of the request. For the customized view implementation we have used a web interface using Java

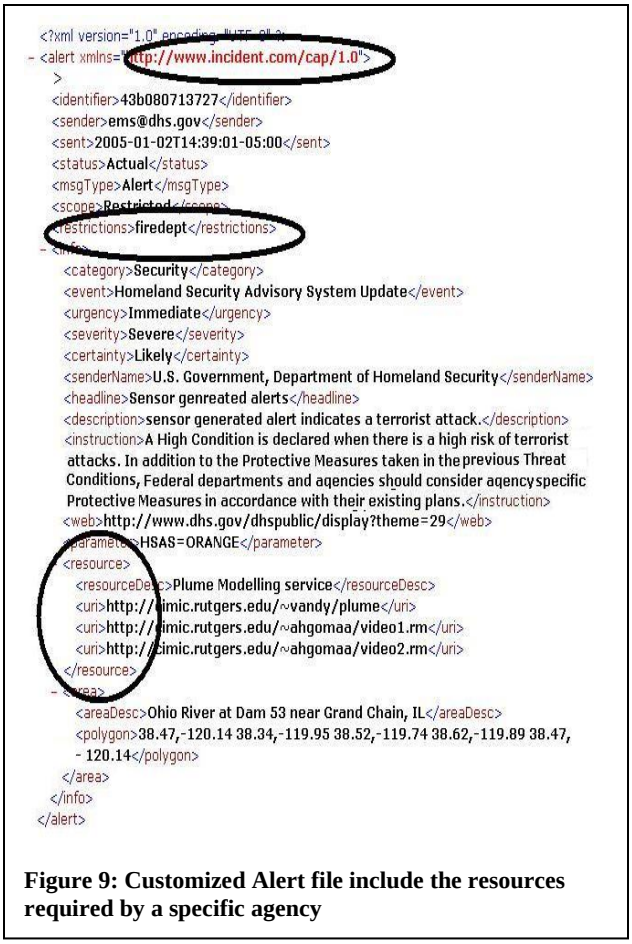


**Figure 9: Customized Alert file include the resources required by a specific agency**

Servlets. The system input is an XML alert generated by the alert generation module. The Alert is associated with location coordinates to be presented on a map in a GIS interface, along with the information to be presented to the Fire department, Police Department and Health Department.

The general view of the alert in the integrated environment is represented as shown in figure 6, whereas the customized view for health agency is shown in figure 7. Figure 8 shows the complete XML specification for the generalized view and figure 9 shows the customized XML for a specific agency.

# 4. RELATED WORK

Alert management goes hand in hand with network event management techniques (such as [HMP02]. It deals with mining of patterns from event datasets. [HMP02] Consolidates the mining of event bursts, periodic patterns and mutually dependent patterns, which are proposed in [MH01A, MH01B]. Event bursts occur in the event of a system failure or a virus attack. Mining event bursts requires: Finding periods where event rates are higher than a specified threshold and mining for patterns common in those patterns. [MH01B] Discusses mining periodic events or p-patterns. To discover these p-patterns, which are repeated occurrences, period lengths are computed and then temporal associations are found between these time periods. Their approach is to compute the event inter arrival times and then test to see if the inter arrival counts exceed the normal expectation. However in certain scenarios such as emergency management there may not be temporal associations between events. Some events may have completely different time stamps but may be related by some parameters. [MH01A] discusses mutually dependent patterns or m-patterns, for mining events which occur together whenever they occur, thus occurrence of one event is also an indication to the occurrence of the other event. However there may be cases where this type of a probabilistic relationship may not exist, but there may be a more semantic relationship.

Alert Management systems primarily focus on screening events, building profiles associated with events and send alerts based upon the profiles and events [G04], it describes a common framework for an alert management system. [FP99] deals with the generation of alarms based on activity monitoring of events such as cell phone transactions. It allocates diminished importance of multiple alarms and quantifies the benefit of timely alarms. However it is evident in emergency scenarios that multiple alarms need to be handled carefully so as not to overlook important situations. At the same a timely alarm may not have as much significance unless it is semantically associated with other alarms.

[PK02] describes an activity monitoring technique for streaming data where the data is assumed to be coming from a process that can be modeled using a linear model. However in an emergency scenario this may not apply as the sources generating the alarms may be different and even the alarms generated by the same source may behave differently thus there may be no relationship between the streaming data. The system may be complex to be modeled in a linear model.

It can be seen in the various examples described that most alert management systems are training based where they require training data for creation of profiles and generating the scores for the alerts. However in emergency management systems due to several complexities such as distributed software components, several agencies and a highly dynamic environment this type of a framework may not apply. Primarily in such environments the training data may not be available. Even if we assume that the training data may be captured, some emergency scenarios may not be reflected in the profiles or the scores, which may let actual alerts slip by and generate high number of false positives.

# 5. CONCLUSION AND FUTURE WORK

In this paper we proposed an *alert management system* (AMS), for reducing false positives and generating *aggregated alerts* from a series of alarms. To accomplish this, we first flagged and eliminated the false positives, then we generated *aggregated alerts* from stream of alarms using clustering and identifying the overlap among clusters, finally we identified dynamic flow alert. Once the alerts are generated we dispersed the alerts on the need-to-know basis using the *customized dissemination module* and using a *GIS module* to display the alerts through a user-friendly interface. For the customized dissemination of the alerts to the right people in the right format we described the application of the National Incident Management System (NIMS) and the National Response plan (NRP) protocols. We implemented these protocols using the Common Alerting Protocol (CAP), which is an XML nonproprietary data interchange format.

The sources of the alerts can be further consolidated such that if there are multiple agencies generating the alarms then the source for the aggregated alert needs to be identified using protocols and policies being followed. As part of our future work we intend to address the issue of source consolidation of an alert.

This can also be extended to the aggregation of the alarms to generate a semantic alert, which is a semantic representation of all the alarms in the alert.

Further the customized view generation needs to address the reconciliation of the protocols for dissemination of the alerts in a scenario where multiple agencies at different levels of hierarchy or from multiple states may be involved. Moreover, we propose to generate customized views not only based on the policies and the user 3Cs, but also based on the semantics of the alerts using semantic ontologies.

## 6. ACKNOWLEDGMENTS

This work is supported in part by the National Science Foundation under grant IIS-0306838. The authors would like to thank Vandy Chopra and Edwin Portscher for the implementation of the prototype system and Dr. Soon Ae Chun and Aabhas Paliwal for early input in the design and implementation of the prototype.

## 7. REFERENCES

[AAAB01] N. R. Adam, V. Atluri, I. Adiwijaya, and S. Banerjee. A Dynamic Manifestation Approach for Providing Universal Access to Digital Library Objects. IEEE TKDE, 2001:13(4).

[AAGA03] Vijayalakshmi Atluri and Nabil Adam and Ahmed Gomaa and Igg Adiwijaya, Self-manifestation of composite multimedia objects to satisfy security constraints, Proceedings of the 2003 ACM SAC, 2003: PP 927-934.

[AJA04] N.R. Adam, V.P. Janeja, V. Atluri, "Neighborhood Based Detection of Anomalies in High Dimensional Spatio-temporal Sensor Datasets" ACM Symposium on Applied Computing, March 2004.

[ARC4] ARC IMS 4.0, ArcView 8.3, http://www.esri.com/

[BEX02] F. Beil, M. Ester, X. Xu. Frequent Term-Based Text Clustering, KDD'02.

[CAP03] OASIS Emergency Management TC, Common Alerting Protocol v 1.0, [OASIS 200402], 12 August 2003.

[EKS97] M. Ester, H. P. Kriegel, and J. Sander. Spatial Data Mining: A Database Approach. In Proceedings of the International Symposium on Large Spatial Databases, Germany, July 1997, pp. 47-66.

[EKSX96] Ester M., Kriegel H. -P., Sander J., and Xu X.. A density-based algorithm for discovering clusters in large spatial databases. KDD'96.

[FP97] Fawcett T. and Provost F. (1997), Adaptive Fraud Detection. Data Mining and Knowledge Discovery, 1(3):291—316

[FP99] T. Fawcett and F. Provost. Activity monitoring: Noticing interesting changes in behavior. In Proceedings on the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1999

[G04] Robert L. Grossman, Alert Management Systems: A Quick Introduction, in Managing Cyber Threats: Issues, Approaches and Challenges, edited by Vipin Kumar, J. Srivastava, Aleksandar Lazarevic, Kluwer Academic Publisher, to appear, 2004

[GAA05] Ahmed A. Gomaa, Nabil R. Adam and Vijay Atluri. Adapting Spatial Constraints of Composite Multimedia Objects to achieve Universal Access; IEEE International Workshop on Multimedia Systems and Networking (WMSN'05}Phoenix, Arizona, USA.

[HMP02] Joseph L. Hellerstein Sheng Ma Chang-shing Perng, Discovering actionable patterns in event data, IBM Systems Journal 2002.

[J04] William Jackson, 9/11 commissioner: Don't wait on intelligence backbone, http://www.washingtontechnology.com/news/1_1/daily_news/25051-1.html, Nov 2004

[JAA04] V. P. Janeja, Vijayalakshmi Atluri and Nabil R. Adam , Detecting Anomalous Geospatial Trajectories through Spatial Characterization and Spatio-Semantic Associations, Proc. of 5th National Conference on Digital Government, 2004.

[KR90] Kauffman and P. J. Rousseeuw. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, 1990.

[L00]Levy, A.: Logic-Based Techniques in Data Integration. Kluwer Academic Publishers, Norwell, MA (2000)

[L05] St.Lucie County, Department of public safety, Emergency operations center, http://www.stlucieco.gov/eoc/index.asp, Last Checked, Feb 2, 2005

[M97]Tom Mitchell, Machine Learning, McGraw Hill, 1997, PP320.

[MH01A] S. Ma and J. L. Hellerstein, "Mining Mutually Dependent Patterns," Proceedings of the 2001 International Conference on Data Mining (ICDM'01), San Jose, CA, November 2001, IEEE, New York (2001), pp. 409–416.

[MH01B] S. Ma and J. L. Hellerstein, "Mining Partially Periodic Event Patterns with Unknown Periods," Proceedings of the 2001 International Conference on Data Engineering (ICDE'01), Heidelberg, Germany, April 2001, IEEE, New York (2001), pp. 205–214.

[OBS00] A. Okabe, B. Boots, K. Sugihara, S. Chiu. Spatial Tessellations: Concepts and Applications of Voronoi Diagrams. pp. 291-410. John Wiley, 2000.

[PK02]Vasundhara Puttagunta and Konstantinos Kalpakis, "Adaptive Methods for Activity Monitoring of Streaming Data" In the Proceedings of the 2002 International Conferences on Machine Learning and Applications (ICMLA'02), Las Vegas, Nevada, June 24 - 27, 2002. pp. 197-203.

[XACML01]OASIS Extensible Access Control Markup Language TC, Extensible Access Control Markup Language (XACML) v1.0 [OASIS 200301]

[XSL03]Extensible Stylesheet Language, World Wide Web Consortium, (Massachusetts Institute of Technology, European Research Consortium for Informatics and Mathematics, Keio University.