# A Multimedia Access Control Language For Virtual and Ambient Intelligence Environments

Bechara AL Bouna
LE2I
CNRS Bourgogne University
IN BP 47870
21078 Dijon CEDEX FRANCE
bechara.albouna@u-bourgogne.fr

Richard Chbeir
LE2I
CNRS Bourgogne University
IN BP 47870
21078 Dijon CEDEX FRANCE
richard.chbeir@u-bourgogne.fr

Stefania Marrara
Information Technology Department
University of Milan
via Bramante 65, 26013,
Crema, CR, Italy
marrara@dti.unimi.it

## ABSTRACT

Access control models are becoming increasingly important in several application domains especially in distributed environments like those addressed by Web Services. Established approaches such as DAC [15], MAC [15] RBAC [11, 12, 22] and others [7, 6, 13, 1] suggest representing users in different ways (labels, roles, credentials, etc.) in order to facilitate the association of authorization and access control policies. In intelligent and virtual ambient applications, users exist in a controlled environment equipped with multimedia sensors such as cameras and microphones, and use their terminals in several application environments. In this paper, we study the problem of integrating multimedia objects into access control models and particularly role-based ones. Here, we describe a Multimedia Access Control Language ($M^2ACL$) in which users and roles are described by using sets of multimedia objects, greatly increasing the flexibility of access control policies and their applicability to virtual and ambient intelligence (AmI) environments. We address potential risks related to the use of multimedia objects by defining the concept of *filter functions* used to aggregate a set of values into a relevant one. Finally, we present a set of functional specification and the experiments conducted to validate the proposed approach.

## Keywords

MultiMedia, Access Control Language

## 1. INTRODUCTION

The rapid evolution of Web-based distributed environments has given rise to a variety of resource protection requirements. The work done by the security research community to address these requirements has led to the definition of a number of access control models, including DAC [15], MAC [15] and others [7,6,13,1]. These models deal with *subjects* whose access to resources is controlled in different ways.

To facilitate associating authorization and access policies to users, early access models gathered users into groups (administrators, managers, etc.) and defined hierarchical links between them. More recently, three basic techniques have been considered in the literature, i.e. *roles*, *credentials*, and *profiles*. In the RBAC [11,12,22] approach, an authorization manager defines a number of roles (to be later assigned to users) and assigns to each role a set of permissions. Roles are connected via hierarchical links to facilitate permission inheritance. While roles work especially well within organizations, credential-based techniques have proved to be effective in the open Web environment, where the same subjects access multiple independently managed services. For instance, in [1], the authors focus on describing subjects based on credentials such as age, country, etc., and put forward the idea of a credential hierarchy, later to be exploited in many practical systems. Similarly, in [6], user profiles have been used to group users using various features (software, device, network, etc.).

In this paper we study the impact of integrating descriptions of subjects with multimedia objects. We describe here a MultiMedia Access Control Language ($M^2ACL$) where users and roles can be defined via conditions on sets of multimedia objects, increasing the expressive power and flexibility of access policies. The need for including multimedia objects is becoming more and more frequent in a number of application fields, including entertainment, health-care, e-government, e-learning, etc. In intelligent ambient applications, users work in a controlled environment equipped with multimedia sensors such as cameras and microphones, and use their terminals (desktop, information kiosks, laptops and handheld devices) to access multiple applications. However, the use of multimedia features in access control models may cause uncertainty in enforcement decisions. We address the problems of uncertainty when multimedia objects are used, even if the risk of false positives is entirely acceptable in several applications. The rest of the paper is organized as follows. In Section 2, we describe some application scenarios motivating our approach. In Section 3, we describe our core model by extending some components of RBAC (Roles, Hierarchy, and Constraints) to handle multimedia objects. In Section 3, we briefly outline the syntax of a language, $M^2ACL$, and discuss the model underlying. In Section 5, we present an application scenario while in Section 6 the experimental results. Section 7 concludes our paper and snapshots some of our future directions.

## 2. MOTIVATING SCENARIOS

We start by describing two sample scenarios showing how the use of multimedia objects can be beneficial in Web-based access control.

### 2.1 Scenario 1

The first scenario deals with the access to a hospital, including a research department with different labs, some of them requiring a high level of security (e.g. because they stock viruses and bacteria). The lab has several surveillance cameras at the entrance doors of each department and in each room of all departments. As detailed below, each lab's staff is divided into several groups depending on the role they occupy and the department they belong to:

- *Lab technician* handling bio samples must wear white suits and grey caps. They have access to storing areas and cold rooms.
- *Researchers* responsible of testing viruses must wear yellow suits with intoxication masks. So dressed, they are allowed to enter testing area and cold rooms.
- *Security guards* supervise access rooms and main hall. In order to make themselves recognizable by other staff and visitors, they must wear red berets and dark navy suits.

In order to enforce the dress code as well access permissions based on identity, the authorization manager needs to define roles on the basis of multimedia objects (as well as on traditional credentials) and to associate appropriate permissions to any combination of multimedia objects and credentials. Once the dress code has been verified, staff member can complete getting their role by using any authentication technique.

### 2.2 Scenario 2

The second scenario is an open-web application loosely in the line of *Second Life*, where an entertainment company organizes an Internet-based aftermath to in-presence parties taking place in a disco. The company wishes to invite to the post-party meeting (with different roles) only people who showed up for the original party. Obviously, there is no directory of names and profiles. A "virtual bouncer" at the door of the Internet party will use social event snapshots (or individual pictures) to assign roles to corresponding people.

- *DJs* wore a baseball cap and a yellow T-shirt with the promotional logo of the virtual disco. In the virtual edition of the party, they have access to backstage areas and virtual hi-fi equipment.
- *Professional dancers* were dressed in costumes. They are allowed to enter the stage and other privileged dance areas.
- *Party-goers* followed no special dress code. They have access to the main hall only.

After verification and role assignment, people are redirected to authorized "virtual rooms" of the Internet party.

## 3. PROPOSED APPROACH

We now define the main concepts needed for a full understanding of our multimedia access control language. In particularly, we introduce *Multimedia Identifiers* and their related components (multimedia objects, multimedia functions, filter functions, etc.) which represent the core of our approach.

### 3.1 Definitions

**Definition** 1 **- Multimedia Object:** allows representing several types of multimedia data such as text, image, video, salient object, speech, etc. Due to its simplicity and its high expression power, here we adopt the MPEG-7 compatible description model provided in [5]. Therefore, a Multimedia Object $Mo$ is represented as follows:

$$Mo : (id_{Mo}, O^*, A^*, F^*, R^*) \quad (1)$$

where:

1. $id_{Mo}$: represents the identifier of the multimedia object
2. $O$: contains the raw data of the object (image, video, or audio) stored as a BLOB file or URI
3. $A$: is the metadata describing the multimedia object. It describes both user- and context-related information and can be written as: $(a_1 : v_1, a_2 : v_2, ...)$ where $a_i$ and $v_i$ represent an attribute and its corresponding value (ex. age: 18, profession: student, etc.)
4. $F$: describes the physical content of a multimedia object (such as Color Histogram, Color distribution, Texture Histogram, shapes, Duration, Audio freq., Amplitude, Band no., etc.)
5. $R$: describes the set of relations existing between multimedia objects [1].

For instance, the following multimedia object: $Mo_1(id = 1, O =' yellow\_suit.jpg', A =' object.name : suit', F =' DominantColor = (14, 24, 20)', null)$ represents an image `yellow_suit.jpg` describing an object "suit" having a yellow dominant color (in the RGB color space) without any link with other objects.

**Definition** 2 **- User**: the set of multimedia items used to describe a subject. As multimedia objects can be used for user description in several applications, we can formalize the user notion as follows:

$$U : (id_u, \overline{\overline{Mo}}^*)) \quad (2)$$

Where:

1. $Id_u$ is the user identifier
2. $\overline{\overline{Mo}}^*$ represents the set of reference multimedia objects describing the user.

[1]These relations can be spatial, semantic, similarity, temporal, etc.

**Definition 3 - Multimedia Function**: is used to handle the comparison between (and feature extraction from) multimedia objects. Many types of multimedia functions have been proposed in the literature and some of them are available in commercial systems. For instance, some of them are provided as extensions to SQL-operators in DBMSs such as Oracle and DB2 [8, 17], while others are accessible via APIs [14, 19] and web services [2] within special-purpose platforms for multimedia data processing. A complete review of such functions and their applications are out of the scope of this paper; the interested reader can refer to [3]. In our approach, we formally write a multimedia function $f$ as follows:

$$f(\overline{\overline{Mo}}^*, Mo^*) \rightarrow (\alpha_f^B)^* \quad (3)$$

where:

1. $\overline{\overline{Mo}}$ represents a *Reference Multimedia Object (RMO)* usually defined by the authorization manager.
2. $Mo$ represents an input multimedia object. It can be provided by multimedia devices of different types such as surveillance cameras, webcams, sound recorders, etc.
3. $(\alpha_f^B)$ is a returned threshold representing the native confidence score of a Boolean value B with respect to the multimedia function. It varies in [0,1] interval.

**Definition 4 - Filter Function**: is applied to the outcome of a comparison to select or compute a single value for decision-making. A filter function can be defined by any probabilistic function such as the combination rule of *Dempster-Shafer theory of evidence (DS)* [9, 20] or of *Bayesian Decision theory* [18]. It is formally written as:

$$\mu(f^*, \varepsilon) \rightarrow \alpha_\mu^B \quad (4)$$

where:

1. $f$ is a multimedia function;
2. $\varepsilon$ is an uncertainty threshold ($\in [0, 1]$) representing the percentage of noise that can affect the result (e.g. lightings, multimedia function precision, multimedia device performance, etc. ). The more $\epsilon$ is high, the more uncertainty increases. If omitted, $\epsilon = 0$;
3. $\alpha_\mu^B$ is the filtered confidence score ($\in [0, 1]$) of a Boolean value $B$ with respect to the filter function.

Filter functions are important because the uncertainty of similarity functions may lead to wrong decisions, potentially granting access to unauthorized users or denying it to legitimate ones. In many applications, however, the risk of false positives is fully acceptable. In our first scenario, for instance, a (partial) violation of the dress code (i.e. because someone wears a white rather than a grey cap), would never conduct to a serious security breach. In the second scenario, the occasional Internet party admission of someone who is an identical twin of a participant to the original real-life event, although undesirable, would not disrupt the scheme. In other cases where security breaches are crucial, filters can be used in conjunction with ordinary credentials to increase access verification [4, 3].

**Definition 5 - Multimedia Condition** allows the authorization manager to handle the analysis of a set of inputs using none or more multimedia functions and a filter function. A condition is `true` iif the result returned by the filter function is satisfied within the Boolean operator and the predefined threshold. A *multimedia predicate* $\overline{P}$ is formally written as:

$$\overline{P} : [Mo|\mu|e]\theta v \quad (5)$$

where:

1. $\mu$ is a filter function;
2. $e$ is an expression;
3. $\theta$ is a multimedia comparison operator containing traditional operators (e.g.$=, \neq, <, >, \leq, \geq$ etc.) and similarity operators used in multimedia queries (such as range [16] and K-Nearest Neighbor queries [10]);
4. $v$ is the validation value. It could be a simple or complex (such as $Mo$)

**Example:** Let us take the first scenario described in Section 2 and show how the authorization manager (AM) can identify *researchers* wearing yellow suits in our approach. We assume that:

1. the combination rule of Dempster-Shafer's theory of evidence (DS) is used as the filter function.
2. two different multimedia functions $f_1$ and $f_2$ are used to analyze multimedia objects:
   - $f_1$: is based on *feature extraction*;
   - $f_2$: is based on *SVM classifiers*

The AM defines the following function contents as:

$$\overline{f_1(\overline{Mo_1(\ldots, O =' suit_{yellow}.jpg', \ldots)}, \overline{Mo_{10}(\ldots, O =' snapshot\_lab.jpg', \ldots)})}$$

$$\overline{f_2(\overline{Mo_1(\ldots, O =' suit_{yellow}.dat', \ldots)}, \overline{Mo_{10}(\ldots, O =' snapshot\_lab.jpg', \ldots)})}$$

where the *RMO* is a suit_yellow image and dat file (or folder) to be compared with an input object called `snapshot_lab.jpg` using the two multimedia functions. He also defines the following multimedia predicate to detect the existence of yellow suits in the picture:

$$\overline{P_1} : \mu((f_1, f_2), 0.1) > 0.8$$

Let us assume that $f_1$ returns a confidence score of 80%, and $f_2$ a confidence score of 60%. These scores represent the probability of validating the concept of detecting the yellow

suit in the snapshot image (`snapshot_lab.jpg`). They are related to the mass functions ($m_i$) defined in the DS theory of evidence. For instance, in our case $m_1(true)$ holds the probability of detecting the yellow suit which is determined using the first function whereas $m_2(true)$ holds the same concept thus determined with the second function. According to the DS theory combination rule, we can then compute the filtered confidence score as follows:

$$m_{1,2}(true) = \frac{m_1(true) \times m_2(true) + m_1(true) \times m_2(\{true, false\}) + m_1(\{true, false\}) \times m_2(true)}{1 - K} \quad (6)$$

where: $K = m_1(true) \times m_2(false) + m_1(false) \times m_2(true)$ represents the conflict in the combination rule. In order to take into account uncertainty when comparing multimedia objects, the uncertainty threshold specified by the authorization manager in the filter function is deduced from the probabilities of the concept to be determined: $m_1(true) = m_1(true) - m_1(true) \times \varepsilon = 0.8 - 0.80.1 = 0.72$ $m_2(true) = m_2(true) - m_2(true) \times \varepsilon = 0.6 - 0.60.1 = 0.54$.

However, if $m_{1,2} = (false)$ is being calculated, (i.e., the fact that the yellow suit is not detected in the input image), the threshold would be deduced from $m_1(false)$ and $m_2(false)$.

- $m_1(\{true, false\}) = 1 - (m_1(true) + m_1(false)) = 0.08$
- $m_2(\{true, false\}) = 1 - (m_2(true) + m_2(false)) = 0.06$

After applying the combination rule, we obtain the following filtered confidence value: $m_{1,2}(true) = 0.786$

Therefore, the application will assume that no yellow suit is identified within the provided snapshot, as the evidence for it lies below the acceptance threshold defined in the predicate.

**Definition 6 - Multimedia Identifier**: allows assigning users to a set of predefined multimedia-based clusters using the multimedia objects that identify them. This technique resembles what happens in traditional approaches (particularly RBAC) where users are assigned to a set of groups or roles using their characteristics (most often on their job functions). A user assigned to a specified *Multimedia Identifier* $M_{Id}$ activates it when the objects identifying the user satisfy the identifier's multimedia predicate. We formally define our $M_{Id}$ model as follows:

$$M_{Id} = (Id_{M_{Id}}, description, \overline{P}^{\theta*}, \overline{State}) \quad (7)$$

where:

- $Id_{M_{Id}}$ represents the identifier of the *Multimedia Identifier*;
- `description` represents its description (e.g., "lab technician in full gear");
- $\overline{P}^{\theta*}$ is a set of multimedia predicates linked together using logical operators (AND, OR, NOT);
- $\overline{State}$) represents its Boolean state (enabled or disabled).

Such representation adds flexibility to classical specifications and approaches, making it possible to define generalized multimedia groups or roles by combining $MId$ with traditional models.

For example, $M_{Id_1} : (1,' Researcher', \overline{P_1}, enabled)$ and $M_{Id_2} : (2,' Lab\ Technician', \overline{P_2}, enabled)$ allow defining two *Multimedia Identifiers* based on *yellow* and *white suits* respectively. Thus, each user wearing a yellow or white suit would be automatically assigned to $M_{Id_1}$ or $M_{Id_2}$ respectively iif $\overline{P_1}$ and $\overline{P_2}$ are valid.

**Definition 7 - Multimedia Identifier Links**: in the RBAC model, the roles hierarchy is usually described as an inheritance relation between different roles which mirrors an organization's chain of commands. This can be readily applied to relations that might exist between different $M_{ids}$ (e.g., between a lab technician in full gear and another one wearing ordinary clothes save for the grey cap and the surgical mask) and extended by additional types of relations (e.g., containment) as well. The link model is formalized as follows:

$$Link : (id_L, M_{Id_{Start}}, M_{Id_{End}}, type) \quad (8)$$

where:

- $Id_L$ represents the identifier of the link existing between the different $M_{Ids}$;
- $M_{Id_{Start}}$ represents a start node;
- $M_{Id_{End}}$ represents an end node;
- `type` describes the type of link established between two $M_{Ids}$. A type could be a hierarchical link, a containment link, a similarity link, etc.

For instance, if $Link_1$ is: $(Id_{L_1}, M_{Id_1}, M_{Id_2},' hierarchical')$, all permissions of $M_{Id_2}$ will be inherited by $M_{Id_1}$ [2].

## 3.2 Constrained $M^2ACL$

In RBAC, constraints support *separation* of duty relations when defining roles. Such relations are used to prevent conflict of interests when managing hierarchical levels of authority. In $M^2ACL$, such relations need to be addressed differently, due to the use of multimedia objects. Two multimedia-specific types of separation duty are defined below:

[2]The link model provides enough expressive power to represent relationships between $M_{Ids}$. Of course, several graphs can be established according to the types of links. In addition, it can be effectively extended to other components of our access control model. However, this issue will not be addressed here.

### 3.2.1 Multimedia Static Separation of Duty (MSSD)

Conflict of interests arises when a multimedia object describing a user is associated to several $M_{ids}$ with conflicting permissions. Such constraints reduce the number of permissions attributed to a user. It defines restrictions for *User/MId assignment*. For instance, consider $M_{Id_i}$, $M_{Id_j}$ and the user $U_1$. Let $M_{Id_i}$ be defined using $I_1$, using $I_2$, and $U_1$ identified by the image $UI_1$. $U_1$ is automatically assigned to both $M_{ids}$ due to the salient objects existence in both $I_1$ and $I_2$ as well as in $UI_1$. In order to prevent conflicts, the authorization manager should be granted the possibility to avoid such assignments. For this reason, we define a multimedia static separation of duty notion to control User/$M_{Id}$ assignment. It is formalized as follows:

$$MSSD(id_{mssd}, id_M Id^*, Card) \quad (9)$$

where:

- $id_{mssd}$ represents the identifier of the MSSD relation;
- $id_{M_{Id}}*$ represents the set of $M_{Ids}$ on which the MSSD relation is applied. This set contains at least two identifiers;
- `Card` represents the cardinality (greater than 1) indicating a combination of $M_{Ids}$ that would constitute a violation of the MSSD relations.

For example, if $MSSD_1$ is defined as: $(1, \{M_{Id_i}, M_{Id_j}, M_{Id_z}, \}, 2)$, this means that users are not allowed to be simultaneously assigned to two of the three $M_{Ids}$.

### 3.2.2 Multimedia Dynamic Separation of Duty (MDSD)

In the RBAC model, dynamic separation of duty is used to prevent users from *activating* simultaneously roles that may arise conflict of interests. In $M^2ACL$, dynamic separations of duty add operational flexibility to the model. For instance, For example, let us consider a user $U_1$ assigned to both `lab technician` $M_{Id_1}$ and `researcher` $M_{Id_2}$. According to the hospital policy, $U_1$ is not allowed to carry on teaching activities (e.g., grade graduate student papers) while he is in full gear, working in the virology lab. Such issue can be addressed using MDSD. In our model, we define a MDSD as follows:

$$MSSD(id_{mssd}, id^*_{M_{Id}}, Card) \quad (10)$$

where:

- $id_{mssd}$ represents the identifier of the MSSD relation;
- $id_{M_{Id}}*$ represents a set of attached $M_{Ids}$. This set contains at least two identifiers;
- `Card` represents the cardinality (greater than 2) indicating a combination of $M_{Ids}$ that would constitute a violation of the MSSD relations.

For example, if $MDSD_1$ is: $(1, \{M_{Id_i}, M_{Id_j}, M_{Id_z}\}, 2)$, this means users are not allowed to activate simultaneously two of the three $M_{Ids}$.

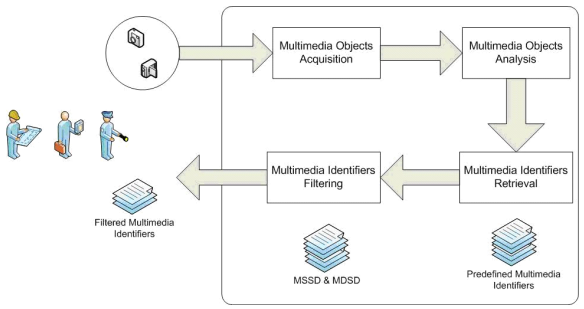


**Figure 1: Architecture of our approach**

## 4. FUNCTIONAL SPECIFICATION

In this Section, we describe the architecture of our system for processing *Multimedia Identifiers* and we point out some of the functions essential for a flexible enforcement of the $M^2ACL$ language. On the basis of such functions, the system decides whether a user $U$ has the right to execute a specific operation by retrieving the set of Multimedia Identifiers to which $U$ is assigned[3].

The main activities performed by our architecture are shown in Figure 1. In particular:

- **Multimedia Identifier Retrieval**: returns the set of Multimedia Identifiers assigned to a specified user identified by the set of multimedia objects seized (or captured). If there are no Multimedia Identifiers assigned to the specified user, the system denies him from accessing the system.

- **Retrieved Multimedia Identifier Filtering**: returns the set of Multimedia Identifiers satisfying the defined constraints and conditions.

The following procedure tests the validity of a given multimedia predicate *based on the input and reference multimedia objects*:

[3]Here, we shall not deal with multimedia object indexing techniques which are essential for our language's enforcement (and, indeed, for any kind of multimedia retrieval) to scale well. Rather, we assume a multimedia data handling layer to provide an indexing facility, which should be easily extensible with new features.

**Algorithm 1**: $Invoke_{\overline{P}}$

**Input**: $\overline{P}$, // *a multimedia predicate*

$\overline{\overline{Mo_j}}^*$, // *the predefined multimedia objects*

$Mo_i^*$ // *the input multimedia objects*

**Output**: *Res // the returned result* $\in \{0, 1\}$

```
1 f[]= get_MF(P̄) // f[]: table of multimedia functions
2 μ = get_FF(P̄)
3 ε = get_FF_UT(P̄)
4 for each(f_i IN f[]) do
5 |  (α^B)* ← f_i(Mo_k*, Mō_j*) // Mo_k* ⊆ Mo_i*
6 end
7 Res = Evaluate_P̄(P̄, μ((α^B)*, ε))
8 return Res
```

The `Invoke_P` procedure takes a $\overline{P}$ , an input multimedia object and the predefined multimedia objects and returns a result $Res \in \{0, 1\}$. In the first three steps, the multimedia functions, the filter function and the uncertainty threshold $\varepsilon$ related to the $\overline{P}$ are retrieved. Each multimedia function is executed against a set of defined multimedia objects $Mo_j^*$ in step 5 taking a set of multimedia objects $Mo_i^*$ as input. The result of the execution is a set of thresholds sent to the filter function. In step 7, the $\overline{P}$ is evaluated based on the returned result of the filter function.

**Multimedia Identifier Retrieval** ($M_{Id}$R): Returns the set of *Multimedia Identifiers* to which a user $U$ (identified by a set of Multimedia Objects) is automatically assigned. It is written as follows:

**Algorithm 2**: $M_IdR$

**Input**: $Mo_i^*$, // *an input Multimedia object taken from multimedia devices*,

$M_{Id}^*$ // *a set of stored Multimedia Identifiers*

**Output**: $Ass\_M_{Id}^*$ // *assigned Multimedia Identifiers*

```
1  for each(M_Id_i IN M_Id*) do
2  |  P̄*=get_P̄*(M_Id)
3  |  for each(P̄_i IN P̄*) do
4  |  |  Mō_j* = get_Mo(P̄)
5  |  |  Result* ← Invoke_P̄(P̄_i, Mo_i*, Mō_j*)
6  |  end
7  |  if (compute(Result, P̄*)) then
8  |  |  Ass_M_Id* ← M_Id_i
9  |  end
10 end
11 return Ass_M_Id*
```

The above procedure retrieves all **Multimedia Identifiers** that a user (identified by a multimedia object) is assigned to. The procedure takes a multimedia object and a predefined set of Multimedia Identifiers as input and returns a set of Multimedia Identifiers with matched $\overline{P}$. For each Multimedia Identifier in the specified set, we first extract predicates and their related multimedia objects. Then, we test the validity of each predicate $\overline{P_i}$ based on the predefined multimedia objects and the seized one. The returned result (containing the results returned by all the $\overline{P}$ defined in the $\overline{P}*$) is computed according to the original $\overline{P}*$. In other terms, logical operators are evaluated depending on the returned results. If the evaluation succeeds the *Multimedia Identifier* is assigned to the table $Ass\_M_{Id}*$.

**MDSD Constraints Validation (MDSD_CV):** tests if users are allowed to activate a stored *Multimedia Identifier* regarding a set of MDSD constraints. We define it below:

**Algorithm 3**: $MDSD\_CV$

**Input**: $M_{Id}$ // a Multimedia Identifier,

$MDSD^*$ // a predefined set of Multimedia Dynamic Seperation of Duty

**Output**: valid

```
1 valid=true
2 for each(MDSD_i IN MDSD*) do
3 |  if get_MDSD(M_Id, MDSD_i) then
4 |  |  valid = false
5 |  |  break
6 |  end
7 end
8 return valid
```

$get\_MDSD(M_{Id}, MDSD_i)$ function tests if an Multimedia Identifier $M_{Id}$ exists in the Multimedia Dynamic Separation of Duty $MDSD_i$.

**Multimedia Identifier Filtering ($M_{Id}F$):** returns the set of *Multimedia Identifiers* valid for a given user $U$ using the $MDSD\_CV$ algorithm defined above. In fact, the algorithm tests if the Multiemdia Ddynamic Separation of Duty ($MDSD^*$) provided as Input and specified for the given user $U$ are valid in order to retrieve all attributed Multimedia Identifiers. It is described as follows:

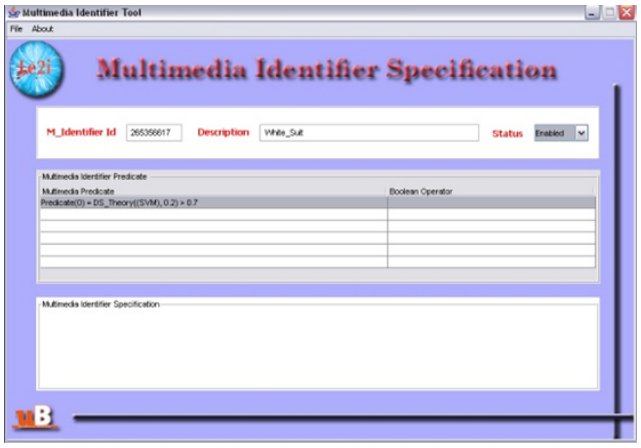


**Figure 2: Multimedia Identifier Specification Interface**

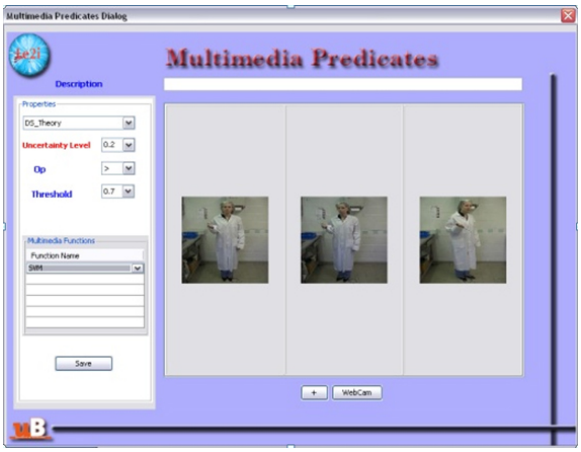


**Figure 3: Multimedia Predicate Specification Interface**

**Algorithm 4**: $M_{Id}F$

**Input**: $M^*_{Id}$, $MDSD^*$

**Output**: $Filtered\_M^*_{Id}$

```
1 for each (M_Id_i IN M*_Id) do
2 |   if (MDSD_CV(M_Id_i, MDSD*)) then
3 |   |   Filtered_M*_Id ← M_Id_i
4 |   end
5 end
6 return Filtered_M*_Id
```

# 5. IMPLEMENTATION

## 5.1 Prototype

In order to validate our approach, we have developed a prototype to ease the definition and the manipulation of $M^2ACL$ *Multimedia Identifiers*. Our prototype provides the authorization manager the possibility to create *Multimedia Identifiers* using graphical components. It is constituted of two different interfaces; one for specifying *Multimedia Identifiers'* simple characteristics (`$M_{Id}$, Description, and Status`) and the other to define related multimedia predicates. In the first interface (see Figure 2) the *authorization manager* defines the *id*, the description and the status of the *Multimedia Identifier* to be specified.

Thereafter, he uses the table to add multimedia predicates which can be defined using our second interface (see Figure 3) and enters the Boolean operators in order to create an expression. For multimedia predicates, the authorization manager is able to specify:

- The filter function and its related uncertainty threshold
- The logical operator (>, <, =, etc.)

```
<?xml version="1.0" ?>
<!-- Multimedia_Identifier_265356617 White_Suit  -->
<Multimedia_Identifier>
  <MI_Id>265356617</MI_Id>
  <Status>Enabled</Status>
  <Description>White_Suit</Description>
- <MPredicate>
  - <Filter_Function name="DS_Theory">
      <M_Function name="SVM" />
      <Level>0.2</Level>
    </Filter_Function>
    <Operator>></Operator>
    <Threshold>0.7</Threshold>
  - <Predefined_Mo>
      <Mo location="C:\White_Suit\CIMG2250.JPG" />
      <Mo location="C:\White_Suit\CIMG2251.JPG" />
      <Mo location="C:\White_Suit\CIMG2257.JPG" />
    </Predefined_Mo>
  </MPredicate>
</Multimedia_Identifier>
```

**Figure 4: $M_{id}$ described using XML**

- The overall threshold to which the result should be compared to.
- The set of multimedia functions and their reference multimedia objects

The set of reference multimedia objects can be acquired using a simple browsing window or using snapshots from live broadcasting Webcam.

After creating the Multimedia Identifier, the authorization manager can also export it into an XML document (see Figure 4).

## 5.2 Application Scenario

Let us consider again our second scenario. It is the case of a research hospital with different labs having a camera at each entrance. The system gives access to its staff upon several


**Figure 5: Examples of Multimedia Identifiers**

objects depending on their *Multimedia Identifiers*, as well as on their credentials. Figure 5 shows the multimedia objects that constitute *Multimedia Identifiers*.

These multimedia objects are described as follows:

- $Mo_1.O =$ `"yellow_suit.jpg"`
- $Mo_3.O =$ `"safety_goggles.jpg"`
- $Mo_4.O =$ `"white_suit.jpg"`
- $Mo_5.O =$ `"grey_caps.jpg"`

$M_{Id_1}$ is activated when users are wearing yellow suits and safety goggles, it is defined as follows (we use the same predicate as defined in the *Multimedia Identifier* section): $M_{Id_1} : (Id_1,' Researcher', \overline{P_1}, enabled)$ where $\overline{P_1} : \mu((f_1, f_2), 0, 1) > 0.8$ ($f_1$ and $f_2$ are two multimedia functions on feature extraction and SVM classifiers respectively).

$M_{Id_2}$ is activated when users are wearing white suits and grey caps. It is described as follows: $MId_2 : (2,' Labtechnician', \overline{P_2}, enabled)$ where $\overline{P_2} : \mu((f_1, f_2), 0, 1) > 0.7$.

Using our model, the authorization manager is able to define the following set of tasks:

1. Defining separation of duty constraints to control *Multimedia Identifier* state operations (activation, assignment, etc.).
2. Assigning to each role a set of permissions described as approval for executing a set of operations upon a set of objects. When a defined permission is specified to a given user, this means that the subject has been granted the possibility to perform an action upon specified objects (resources to be protected from unauthorized access). These objects can be of different types such as room access, web applications, multimedia objects access (images, video scenes, etc.) depending on the domain of application. Both permissions and objects definition is out of the scope of this paper. These permissions can be defined as:
   - $Permissions_1^* \rightarrow M_{Id_1}$
   - $Permissions_2^* \rightarrow M_{Id_2}$
3. Creating a hierarchical link between *Multimedia Identifiers* $A_1$ and $A_2$ to preserve the authorization line of authority between lab technicians and researchers. The link is defined as: $Link(001, M_{Id_1}, M_{Id_2},' hierarchical')$ Where 001 is the id of the link, $M_{Id_1}$ and $M_{Id_2}$ represent the *Multimedia Identifiers* of the start and the end node. The defined link permits the inheritance of permissions between $M_{Id_1}$ and $M_{Id_2}$. In other terms, $M_{Id_1}$ (researchers) inherits automatically the permissions assigned to $M_{Id_2}$ (lab technician). This way, $MId_1$ has a set of permissions assigned by the authorization manager and a set of permissions inherited from $M_{Id_2}$ due to the hierarchical link existing between both *Multimedia Identifiers*.

When a user $U$ arrives at the entrance of the department, a picture (or several ones) of her will be captured by a surveillance camera in order to detect to which *Multimedia Identifier* she should be effectively assigned.

# 6. EXPERIMENTS

In this Section, we describe the set of experiments conducted to test the relevance of our approach. The tests were made on a Pentium IV 1.7 GHz with 512 MB RAM and using a USB Webcam. Our database contains snapshots of objects which form the Multimedia Identifiers to be created and the followed instance images. We used a multimedia function based on color object recognition and a SVM classifier. This later computes decisions based on a set of classes representing the trained images (See [21] for more details). The set of experiments we performed are divided into several categories:

- Filter Functions Relevance: permits to test the relevance of filter function regarding a determined error ratio from multimedia functions.
- Filter Functions Evolution: reveals the evolution of filter functions computed with an increased value of uncertainty threshold.
- Execution Time: allows to detect the processing time to compute a final decision from the filter function.

## 6.1 Category 1: Filter Functions Relevance

The objective behind this test is to calculate the error ratio rose when detecting a person wearing different objects in order to validate the relevance of filter functions. In fact, we have trained 10 different classes representing objects to be recognized. The list of classes is described in the following:

1. Black_Sweater
2. Black_TShirt
3. Blue_TShirt
4. Brown_Jacket
5. Brown_Sweater
6. Gray_Cap
7. Green_Sweater
8. White_Shirt

| Class N° | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Error Rate** | 0.24 | 0.44 | 0.67 | 0.35 | 0.96 | 0.01 | 0.38 | 0.66 | 0.14 | 1.0 |
| **DS Result** | 1.0 | 1.0 | 0.01 | 1.0 | 0.01 | 1.0 | 1.0 | 0.01 | 1.0 | 0.0 |
| **Avg Result** | 0.71 | 0.55 | 0.36 | 0.62 | 0.13 | 0.9 | 0.6 | 0.37 | 0.79 | 0.09 |
| **Min Result** | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 |
| **Max Result** | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 | 0.09 |

**Figure 6: Relevance Results computed with different filter functions**

9. White_Sweater
10. White_TShirt

We have taken 100 images representing people wearing the same object for each of the described classes. These snapshots were taken in different environmental conditions (lightings changes, different positions, etc.). Consequently, for each input image, the SVM function returns a threshold related to the relevance of the object detected. The error ratio is calculated based on the number of times the SVM function invokes a decision different than the supposed one. Thereafter, to validate the advantage of using filter functions, we compute the returned results from each input image to several filter functions (DS, Avg, Min and Max) in order to get a unique threshold representing the relevance of the decision. We obtained the following results:

The error ratio is calculated for each of the classes based on 100 input images (representing each class) taken in several environmental conditions which lead to a divergence in the result when detecting the corresponding class. The calculated error ratio depends actually on the relevance of the multimedia functions used, the environmental conditions and users' behavior. This means that the values determined above might vary if one of the latter conditions changes. Nevertheless, what we are interested in is the result returned by the filter functions that are supposed to reflect the final decision (whether a given user is assigned to a *Multimedia Identifier* and has the right to activate it). As we can see in Table 6.1, the Min and the Max functions return continuously the minimum and the maximum thresholds detected from within the 100 thresholds returned which is (0.09 and 0.9). As for the Avg filter function the result depends on the values of the thresholds returned. Thus, let us take the case of class number 2 which is the Black_TShirt. In this case, the Avg function has computed a result equal to 0.55 which considered low regarding the fact the input images represent the class to be detected. In other terms, a value of 0.55 might reduce the potential risk of access violation (error ratio = 0.44) thus decease "information availability". On the other side for the same class, the DS has computed a value of 1.0 which means that the system is certain that the overall decision should be affected to the class Black_TShirt even if the error ratio is relatively high (0.44). Now let us take the example of class number 3 which is the Blue_TShirt. The error ratio determined is equal to 0.67 which is relatively high. In other terms the system is effectively certain that from the 100 input images provided there are at least 67 that doesn't correspond to the class Blue_TShirt. The DS filter function computes a value equal to 0.01 (less than the value of the Avg function) indicating the final decision.

## 6.2 Category 2: Filter Functions Evolution

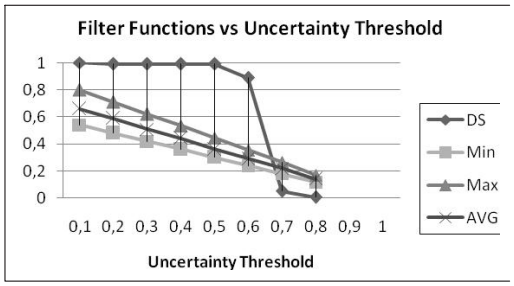


**Figure 8: Computed Thresholds vs. Uncertainty threshold**

This test is elaborated to determine the evolution of filter functions regarding an increased number of uncertainty thresholds; we invoked multimedia functions with several snapshots of the object to be detected. All returned thresholds are computed with different filter functions such as Max, Min, Average, and DS and the returned result is compared to the one specified by the authorization manager (see Table 6.1.

The objective of these tests was to detect whether a certain user is wearing a grey cap or not. We provided the multimedia function a set of 50 input images in which the person is wearing a grey cap with different lighting conditions. The result varies depending on the environmental conditions and the human related behavior (lighting, user positioning, etc.). It returned a set of thresholds between 60% and 90% shown in the following graph (Figure 8).

Figure 8 shows that for a given interval of uncertainty thresholds between 0.1 and 0.5, the DS-Theory has conserved its result. However, with an uncertainty threshold = 0.1, the Min function has returned a 0.54 value which is considered relatively low according to the fact that the user was wearing a grey cap and this might lead to an unnecessary denial of access. Anyway, the returned thresholds depend directly on the multimedia function used and the objective behind defining *Multimedia Identifiers* (precaution-based or access control). When using different multimedia functions results might vary depending on the functions' precision. Thus, for highly secure environments, the authorization manager could use the Min function in order to control at maximum his platform.

## 6.3 Category 3: Execution Time

Here, we determine the execution time taken to compute a final decision from the filter functions. Nevertheless, the process time of multimedia functions is not taken into con-

| $\varepsilon$ | **0.1** | **0.2** | **0.3** | **0.4** | **0.5** | **0.6** | **0.7** | **0.8** | **0.9** | **1.0** |
|---|---|---|---|---|---|---|---|---|---|---|
| **DS** | 1.0 | 0.99 | 0.99 | 0.99 | 0.99 | 0.89 | 0.05 | 0.0 | 0.0 | 0.0 |
| **MIN** | 0.54 | 0.48 | 0.42 | 0.36 | 0.3 | 0.24 | 0.18 | 0.12 | 0.06 | 0.0 |
| **MAX** | 0.8 | 0.71 | 0.62 | 0.53 | 0.44 | 0.35 | 0.26 | 0.17 | 0.08 | 0.0 |
| **AVG** | 0.66 | 0.59 | 0.51 | 0.44 | 0.36 | 0.29 | 0.22 | 0.14 | 0.07 | 0.0 |

**Figure 7: Computed Results from different Filter Functions**

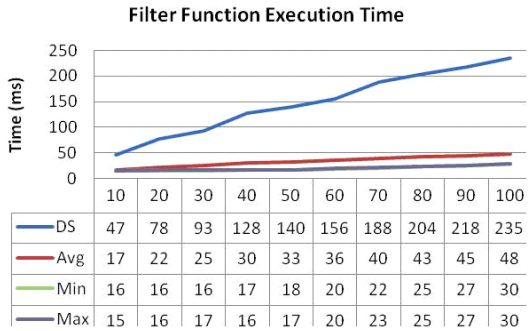


**Figure 9: Execution Time**

sideration which is related to the multimedia function used. In order to calculate the execution time of filter function, we have provided 10 input thresholds for each step to different filter functions (DS, Avg, Min, Max) and after we determined the time computed by each of the functions. The result is shown in Figure 9:

In essence, it shows the time evolution regarding a number of input thresholds. As we can see, the DS filter function has the highest time values for an increased number of thresholds. Nevertheless these values should not affect the final decision invocation time due to its low estimation against multimedia time process.

# 7. CONCLUSION

In this paper, we have presented $M^2ACL$, a multimedia access control language based on the RBAC model, where users and roles can be defined using multimedia objects. Our approach extends the notion of roles to allow assigning users to a set of predefined multimedia based clusters (known as *Multimedia Identifiers*) by analyzing the multimedia objects that identify these users. Nevertheless, the use of multimedia objects within access control models creates potential risks of uncertainty when computing decisions. This is why we defined the concept of filter functions to reduce the possibility of computing unauthorized decisions. We provided an adaptation of the combination rules used in the theory of evidence, in order to aggregate the set of values into a relevant one. We also presented a set of functional specifications to ease the implementation of the defined model and we described our prototype that assists the authorization manager in creating *Multimedia Identifiers*. Moreover, we discussed a set of experiments targeting filter functions relevance and evolution while computing an increased number of uncertainty thresholds. We also analyzed the processing time during filter function evaluation. Several directions need to be explored. For instance, we plan to extend the concept of easy migration from RBAC model into ours in order to assist authorization managers to define hybrid roles, textual and multimedia-based.

# 8. REFERENCES


[1] N.R. Adam, V. Atluri, E. Bertino, and E. Ferrari. A content-based authorization model for digital librarie. *IEEE Transactions Knowledge and Data Engineering*, 14(2):296–315, 2002.

[2] ASP Alliance. http://aspalliance.com/404_image_web_service/, 10 2006.

[3] C. Agostino Ardagna, E. Damiani, Sabrina De Capitani di Vimercati, and Pierangela Samarati. Towards privacy-enhanced authorization policies and languages. In *DBSec*, pages 16–27, 2005.

[4] B. Al Bouna, R. Chbeir, and J. Miteran. Mca2cm: Multimedia context aware access control model. In *Intelligence and Security Informatics, IEEE International Conference on Intelligence and Security Informatics ISI*. ISI, New Jersey, USA„ 2007.

[5] G. Chalhoub, S. Saad, R. Chbeir, and K. Yetongnon. Towards fully functional distributed multimedia dbms. *Journal Of Digital Information Management (JDIM)*, 2(3):116–121, September 2004.

[6] E. Damiani, Sabrina De Capitani di Vimercati, Eduardo Fernández-Medina, and Pierangela Samarati. Access control of svg documents. In *DBSec 2002*, pages 219–230, 2002.

[7] E. Damiani, Sabrina De Capitani di Vimercati, and Pierangela Samarati Stefano Paraboschi. Securing xml documents. In *EDBT 2000*, pages 121–135, 2000.

[8] QBIC DB2 Image Extenders. http://wwwqbic.almaden.ibm.com/, 12 2006.

[9] A. P. Dempster. A generalization of the bayesian inference. *Journal of Royal Statistical.*, Society(30):205–447, 1968.

[10] H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, and A. El Abbadi. Approximate nearest neighbor searching in multimedia databases. In *In Proc of 17th IEEE Int. Conf. on Data Engineering (ICDE), Heidelberg,Germany*, pages 503–511, April 2001.

[11] D. F. Ferraiolo, J. F. Barkley, and D. R. Kuhn. A role-based access control model and reference implementation within a corporate intranet. *ACM Transactions on Information and System Security*, 2(1):34–64, 1999.

[12] D. F. Ferraiolo and et al. Proposed nist standard for role-based access control. *ACM Trans. Information and System Security (TISSEC)*, 4(3):224–274, 2001.

[13] J. Joshi, R. Bhatti, E. Bertino, and A. Ghafoor. Access-control language for multidomain environments. *IEEE Internet Computing*, 8(6):40–50, 2004.

[14] Efg2 Computer labs. http://www.efg2.com/lab/library/imageprocessing/, 10 2006.

[15] C. E. Landwehr. Formal models of computer security. *ACM Comput. Surv.*, 13(3):247 – 278, September 1981.

[16] S. Nepal and M.V. Ramakrishna. Query processing issues in image (multimedia) databases. In *15th International Conference on Data Engineering (ICDE'99)*, page 22, 1999.

[17] Oracle Technology Network. http://www.oracle.com/technology/products/intermedia/, 09 2006.

[18] D. Poole. Logic, knowledge representation, and bayesian decision theory. In *In Proceedings of CL-2000*, pages 70–86, 2000.

[19] Java Community Press. http://jcp.org/en/jsr/detail?id=135, 11 2006.

[20] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[21] F. Smach, C. Lemaitre, J. Miteran, J. Gauthier, and M. Atri. Colour object recognition combining motion descriptors, zernike moments and support vector machine. In *Proceedings of IECON'06, IEEE, Paris-CNAM, France*, pages 3238–3242, 2006.

[22] J. Wang and S. L. Osborn. A role-based approach to access control for xml databases. In *SACMAT 2004*, pages 70–77, 2004.