1. **PROPOSED SOLUTION**

OrBAC is a framework to specify security policies

The next figure represents the General Architecture of OrBAC implementation, this first approach describes the different components and their interactions, and thus, it brings services to the engine.



PyOrBAC Architecture

The idea is to separate in small modules the different functions that integrate the OrBAC engine, in order to simplify its maintainability and performance.

This engine can interact through API with other systems or different implemented user interfaces

**Management Service:**

This module is in charge of OrBAC Administration, Security and Configuration.

Administration (AdOrBAC): regards OrBAC database backups and restore, jobs schedule, replication information, etc.

Security: gives access to the OrBAC modules

Configuration: setting the variables that determine the behavior of OrBAC

**Connection service:**

It is an API that allows the interaction between OrBAC and graphic interfaces or remote systems

**Storage Service:**

The function of this module is to store all the OrBAC elements, as well as Logs and Security Policies. It is in charge of storing all the information related to the elements, policies, configuration settings, logs, etc.

**Context Manager:**

In this component, it must be defined the rules, which are the conditions that indicate that a context is activated, and search for the policies that must be applied and executed.

**Context Agent:**

Some systems are not able to manage all the context defined in OrBAC, so the Context Agent monitors the environment to check if the context has changed, in this case, it informs to the Context Manager the new condition, in order to supply the new policy.

**OrBAC Processor:**

This module is in charge of validating the syntax of the OrBAC commands like creation of organization, roles, activities, object, etc. It also verifies the conflicts among policies; and it translates abstract level policies into concrete level.

**Configuration Agent:**

It translates the policies from the XML format to the set of instructions needed to apply the rules and to execute them

**Transaction Manager:**

It must check that all the configurations related with policy creation or modifications were done in the corresponding targets.

**Synchronization Service:**

It verifies that OrBAC rules are synchronized with the system target configurations.

**Mapping Service:**

This service maintains the relation between real elements and its OrBAC representation. It services can be used during the translation from OrBAC policies to configuration instructions. OrBAC Actions and Objects must have an equivalent in real life.

Two types of entities exists those that can be created automatically and those that not.

1. **XOrBAC**

An OrBAC Target System is a program that uses OrBAC as security policy administrator.

The communications with external OrBAC components are done thought XML files, the definition of these files was established in XML Schema and it is referred in this document as XOrBAC.

XML Schemas are used in order to define the legal construction blocks of the XML documents that will be used to exchange information between the OrBAC System and OrBAC Target Systems. The following paragraph presents the advantages of XSD over DTDs:

XML Schema became a W3C Recommendation in May 2001, it supports data types that are easier to describe allowable document content and validate the correctness of data. It is extensible to future additions, allowing the reuse of other Schemas. It creates data types derived from the standard types and reference multiple schemas in the same document. It supports namespaces. XML Schemas are written in XML format whose benefits is that programmers do not have to learn a new language and XML libraries can be reused to parser Schemas files.

[http://www.w3schools.com/schema/schema\_intro.asp]

1. **XOrBAC STRUCTURE**

XOrBAC Schema is a translation of elements, relation and rules proposed by Alexander Miege.

XOrBAC Schema is divided in several XSD files, with the intention of allowing the maintainability and readability, and reusability of the proposed structures. This separation was done according to the function of each structure, for example, all the data structures related with hierarchies are together in the same XSD file.

Entity type:

This basic data structure allows identify entities and its characteristics, this structure is composed by :

Attribute:

[2] “Attributes enable us to express further information about the entities involved in a security policy.” … “Attributes will be used to design contexts” … “In Or-BAC every object may have some attributes.” … “The definition of attributes depends of the application domain of the security policy.”