Exercise 2 – Service Oriented Architectures

A new Internet company (MyShow) wants to provide a mash-up service where registered users can get the list of movies played in theaters located nearby their homes. After user log-in, the service will provide a list of available theaters whose address is at maximum 1km far from the user's zip code (specified in the registration form). By selecting one of the listed theaters the user can get access to the list of movies shown today in the selected theater, with the associated information such as the movie name, the show times, etc.

To provide the nearby theaters, MyShow exploits a third party movie service (MovieDB) that for each theater of a given city, provides the list of all movies included in the playing schedule. Each movie descriptions contains the movie Name, the movie Rating, and the movie Running Time. The same service also provides information about theaters including the theater Name and the surface mail Address, and information about movie Show Times.

The database lying under the informative system of the movie service is structured as follows:

Theater (idTheater, Name, Address, ZipCode)

Movie (idMovie, Name, Rating, RunningTime)

Shows(idTheater, idMovie, ShowTimes)

We want to design the information system of MyShow. Design and implementation are organized according to increasing levels of complexity.

Level 1 - Design of the user authentication system.

Users will be allowed to use MyShow if and only if they can provide a correct user name and password. Supposing that the user database is organized as reported below, we want to design the user authentication page, and the welcome page where users can ask for the list of available theaters by clicking on the "get Theaters" button.

User (<u>username</u>, password, zipCode)

Level 2 – List of Theaters in XML

By clicking on the "get Theaters" button, MyShow performs a call to the getMovieInformation.php web page offered by MovieDB, which takes as input (POST) the zip code of the user home and provides back a list of theaters formatted in XML. After receiving the MovieDB response, the welcome page of MyShow parses the received XML code and prints out a list of available theaters.

Level 3 – XML query for Theaters

We want to modify the page developed in level 2, for sending the getTheater request in XML. The XML sent to MovieDB must conform to the theaterRequest.xsd schema. Whenever the MovieDB service receives an XML request, it first validates the request and then it provides back an XML formatted list of theaters (theaterResponse.xsd). The welcome page of MyShow validates the XML response and shows the available theaters as a set of list items.