Semantically Enhanced, Multifaceted, Collaborative Access to Cultural Heritage

Project IST-034984

D5.2 MOSAICA Application

Deliverable

Editor: Idea
Contributors: Idea, XLAB
Dissemination Level: Public
Version: 1.1
Date: 29 Nov 2008

Executive Summary

Document Objective

This document describes the concepts behind the application architecture, the ideas that led to the specific design and the methods that have been taken in order to achieve it.

It also describes the various facilities that the application offers and their use.

Targeted Audience

MOSAICA Consortium

Prerequisite

D5.1 MOSAICA System & Application Specification

© Norbert Benamou MOSAICA

This document contains proprietary information of MOSAICA and shall not be reproduced or transferred to other documents, disclosed to others, or used for any purpose other than that for which it is furnished, without the prior written consent of MOSAICA.

The trademark and service marks of MOSAICA, including the MOSAICA mark and logo, are the exclusive property of MOSAICA, and may not be used without permission. All other marks mentioned in this material are the property of their respective owners

Table of Contents

1 Introduction 4

2 BLS – Business Logic Server 6

2.1 BLS Basics 6
2.2 State Manager 7
2.3 Workflow Manager 8
2.4 Interfaces 10

3 User Interface 11

3.1 Overview 11
3.2 General Guidelines 12
3.3 Technology 13

4 User Management 14

4.1 Layers 14
4.2 Entities 14
4.3 Workflows 19

5 Online Communication 22

6 Appendix: List of 3rd Party Tools 23

1 INTRODUCTION

This document summarizes the work and solutions of task T5.4 *Integration of components,* and part of task T5.5 *Application development.*
Aiming to create a project that will rely on cutting edge technology and a state of the art design, the MOSAICA project was envisioned and later actually planned to be based on true software engineering standards, some of which would not be normally implemented in ordinary software products.
In most of the contemporary software projects, design issues are being compromised in order to gain more rapid development (and thus a faster time to market) and reduce the complexity of the implementation.
In the MOSAICA project such structure has been adopted that should allow the followings:

- Rapid development of new features.
- Easy way to add/remove/manage workflows in the application.
- The ability to change feature in the application without any affect on other elements.
- The ability to become (in the future) a big enterprise with worldwide deployment of multiple servers.

The application is roughly divided into 2 layers, the Frontend and the Backend.

The frontend is responsible for all the interaction with the user:

- Navigation within the various screens and between different screens.
- Host frame for MOSAICA autonomous components (i.e. Semantic layer, GIS)
- Initiation of workflows – every process starts with some action that the user does (like clicking on a button or on a link)
- Management of forms that the user has to fill out or update
- First level validation of input data
- Interaction with the backend
- Frontend session management

The backend is responsible for the following:

- Integration between the autonomous components (i.e. Semantic layer, GIS)
- Backend session management
- Running of workflows
- Services that handle application's business logic

If we look at the application from the functional perspective, we identify the following modules:

- Application main frame – Including caption, logo, title, general utilities, menu, toolbars and links to legal and help documents
- User management utilities
- Online communication facilities

To sum it up, MOSAICA application is a set of components that operate harmonically and interact with each other through the tools that the application framework provides for them, using cutting edge technology and a unique design.

2 BLS – BUSINESS LOGIC SERVER

The MOSAICA application has been designed as a set of components that work autonomously and interact with each other only fi they need the services of another component.
The interaction between the various components however, is not always that simple because many times a sequence of interactions is needed in order to perform a single task. Moreover, many times various values have to be maintained along several independent calls.

For all the above purposes, the Application server or the Business Logic Server (BLS) has been designed.
The BLS provides the following services:

- Integration between the various components of MOSAICA.
- Workflow management and execution.
- Session management.
- State handling.
- Clustering ability
- Error handling

Ultimately, provided that all the interfaces between the various components and the BLS are defined, creating a workflow should be as simple as writing a sequence of instruction in a script.
As a side effect to the application integration, the BLS should also provide the followings:

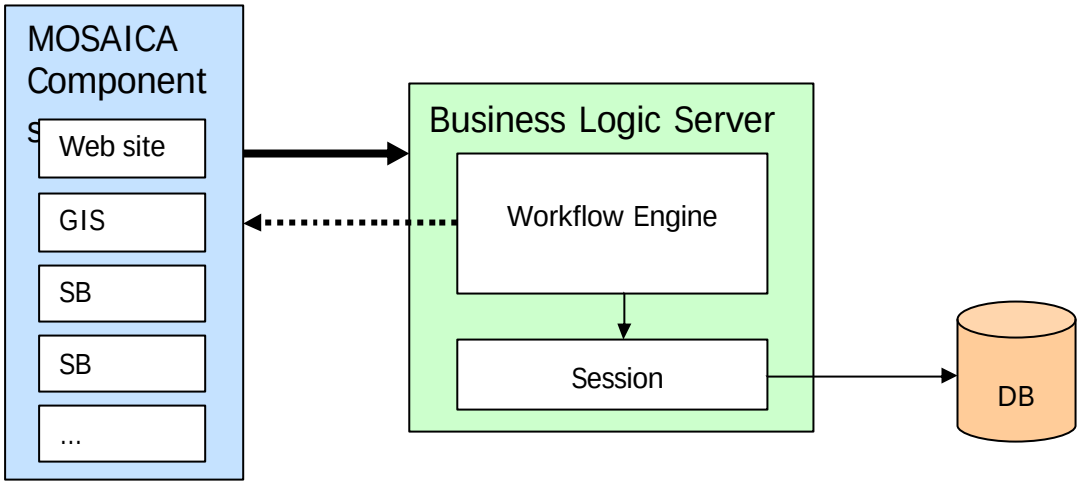
- High performance of application
- Error handling

Clustering and load balancing capability enables the installation of BLS servers on many locations around the world while they would still interoperate and thus enable linear expansion of the MOSAICA project.

2.1 BLS Basics

The Business Logic Server (BLS) has been developed in order to tie together the various components of MOSAICA project.
Being the module that runs the application logic, a lot of effort has been taken in order to make it run efficiently and be able to recover from various errors and problems, such as network failure, power down, and so on.

The BLS comprises of 2 main components: The state manager that is a stateless component that preserves values per user session, and the workflow manager which is essentially a state machine that carries out a sequence of steps (functions) per state and action.



In a more technical manner, to the outside world, the BLS is a web service that runs as part of an application server.
It is a very thin wrapper that utilizes an internal stateless EJB component.
The service itself is of type 'XML Document', which means that all the description of the actions to be taken, are embedded within the XML document and so there is no need of redeployment of the interfaces whenever some of them had changed.
Also, between the various modules of the BLS (App. Server, Workflow manager, State manager) web services are used as interfaces, which enable the following:

- Modularity – One can change or update one part of the system without effecting the other parts.
- Decentralization – The modules can be run from different machines and still function as one unit.

2.2 State Manager

The state manager holds and manages a server side session for each user that is currently using the system.
Within the session, any amount of data about the user can be maintained.
Ultimately, the data is being kept in an SQL database and is being cashed into the system on a 'Most Frequently Used' basis.
In order to prevent 'Garbage' data from being accumulated in the server (usually the result of session that have not been terminated normally), every session is timed out after a predefined elapsed time of inactivity.

2.3 Workflow Manager

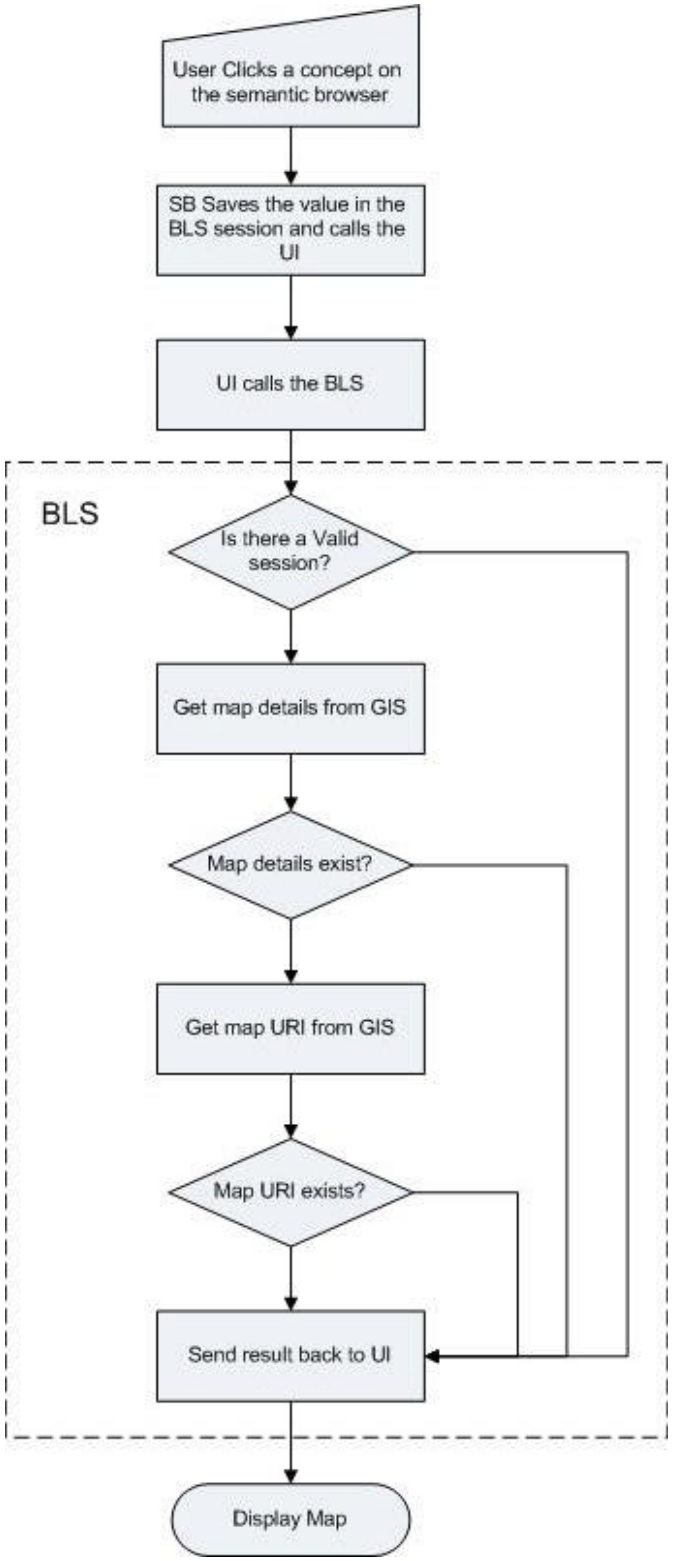
As discussed before, the system comprises of several different components that operate and perform their functions autonomously. Still, sometimes a component has to communicate with another component in order to pass control of the application or get some results. In such cases, the components would have intercommunicated through web services or the application front end would have initiated a call to one of them and then to the other.

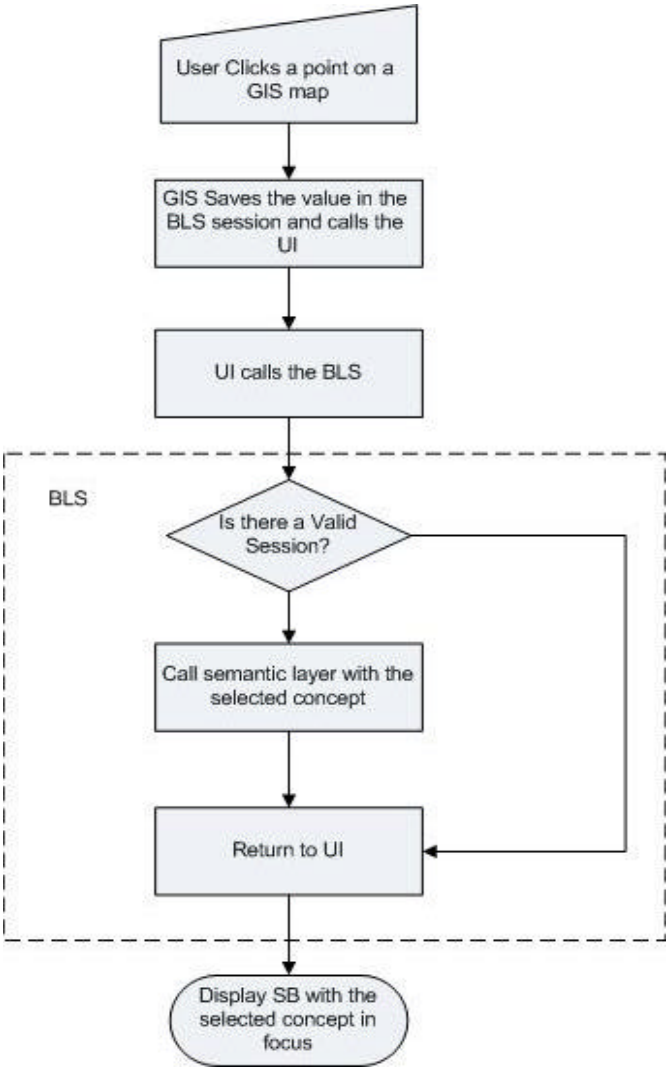
In some cases a series of calls have to be made in a predefined order, which forms a workflow.

The BLS handles the communication between the various components of the application and one of its tasks is to handle the sequence of calls when the control passes from one component to the other.
The requirements for the project stated that the workflows should be configurable without the need to make any changes to the code, so we have developed a module that can run workflows defined in XML files (all the definitions of the workflow are included in the XML file and can be updated at any time). In order to do that, we have adopted a part of an open code project named BPEL and built a module around it.
For the sake of uniformity, many of the internal calls of the BLS are also governed by the workflow manager.

So far only 2 external workflows (workflows that run calls between application components) have been implemented, but it is very easy to implement any number of additional workflows and no additional coding is required for that.
The two workflows are as follows:

1. Semantic browser to GIS.
   The workflow is initiated when a concept is being selected on the browser and the user selects to open a map with all the annotations related to that concept.
2. GIS to Semantic browsing.
   The workflow is initiated when a point is being clicked on the GIS.


User Clicks a concept on the semantic browser
SB Saves the value in the BLS session and calls the UI
UI calls the BLS
BLS
Is there a Valid session?
Get map details from GIS
Map details exist?
Get map URI from GIS
Map URI exists?
Send result back to UI
Display Map



Some of the internal calls are also implemented as a workflow.
For instance the wfGetState call which should return a pointer to all the session parameters.

2.4 Interfaces

For simplicity and security reasons, all BLS interfaces are of type Document, which means that BLS uses only one physical interface, in which all functionality is encapsulated. The required method, incoming parameters, return values and other properties, are all encapsulated within one standard XML document. So in fact, the BLS uses only one service and interprets it internally to address the actual required service.

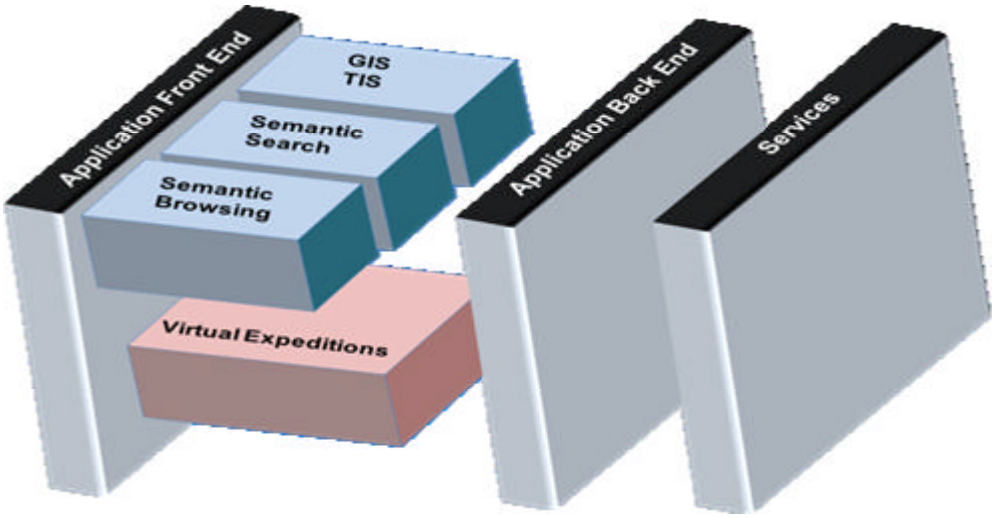
3 USER INTERFACE

3.1 Overview

The MOSAICA web front end acts as a frame, within which reside MOSAICA's different components, which may or may not have their own GUI. These components are the Semantic Components (Semantic Search, Browsing, Resources) and GIS/TIS Components. The various components run their own autonomous, mutually independent (no cross-site linking) GUI and connect with the application framework only when they need some general services (like user management functionality for instance) or if they need to interact with other components, in which case the application serves as a base for the interaction with the BLS, as the web application programming dictates. It then follows that these components are glued together via Business logic server, which orchestrates the internal state of the GUI components and the application. To put it in a simple way – the application front end does not interfere with the GUI management of the various MOSAICA components, but it does serve as a central hub that binds all the components together, defines the location and space allocated for every component on the screen and helps the component to interact with the BLS that serves as the central point for integration between the components on the backend.

Apart from the above mentioned, the application frontend also handles the following functionality:

- Login to the application – Described in more details at the User Management section.
- Account management – Creating, Activating and updating of user accounts - Described in more details at the User Management section.
- Authorization management – Creating, updating and assigning of authorization groups - Described in more details at the User Management section.
- Instant messaging between users of MOSAICA incorporated into the web interface. This feature enables all the users who are currently logged into MOSAICA to interact with each other through an instant messaging system.
- Managing the preferences for the components (Semantic/GIS).
- General information documents – 'Who are we', 'Welcome to MOSAICA'
- General legal documents – 'Terms of Service', 'Copyright/IP Policy', 'Privacy Policy', 'Fair use and Multiculturalism'
- 'Contact Us' information
- Context help for all the application modules.



3.2 General Guidelines

When designing the graphical user interface, the general idea that the result should be a likeable, streamlined GUI that runs independently of the platform used by the user. The following guidelines have been considered while designing the GUI:

- The application should run on every browser.
- The technology that serves for the building blocks of the application GUI should be used free of charge and should be a cutting edge, state of the art technology.
- The interface should be friendly and as much as possible self-explanatory for any user that wants to use the system.
- The interface should incorporate elements that are customary by Web2 standards (no complete frame reloading, desktop like elements)

JavaScript helps break the barrier between the static pages and the interactivity for the users. To ensure an efficient development process and support for a wide range of the web browsers, Google Web Toolkit with applicable extensions has been chosen as the development platform.

Although MOSAICA application is targeted among others to the general public and The GUI provides general access to occasional visitors, a wide and rich functionality is also implemented to handle the registered users and their accounts' life cycle. This includes the formation of an account when a user registers, the taking part in a variety of groups, and setting up and auditing the activities of the users' children. The account

management enables tailoring the user's experience of the MOSAICA content to the user's needs. This part is discussed in details at the User Management section.

3.3 Technology

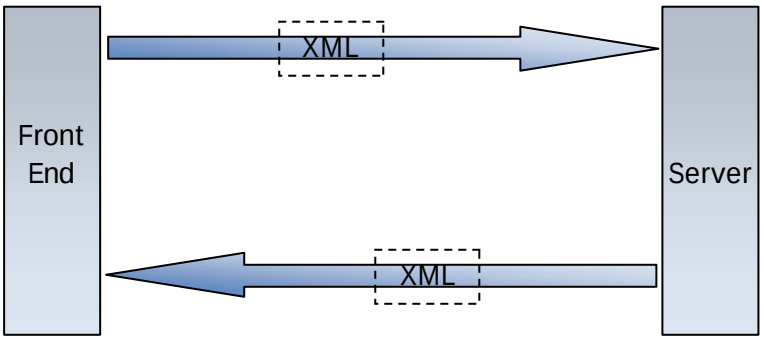
- Google Web Toolkit (GWT) serves as the basis for developing the graphical user interface (GUI). This provides the ease of developing the graphical user interface in Java, while at the same time results in a streamlined and responsive web interface.
- GWT-EXT serves as an extension of the basic GWT components, containing a wider variety of pre-built components to be used in the GUI.
- Apache/Tomcat serves the web application for the users' web browsers.
- XMPP instant messaging protocol, based on the OpenFire server. It also includes the Java XMPP bindings provided by the Smack and SmackX libraries for use from Java.
- Axis (and REST WS): Axis framework is used for calling the web services (WS). More importantly, the Axis and our implementation of the REST WS is used for coordination of the underlying distributed components, which are used in MOSAICA.

4 USER MANAGEMENT

The user management (UM) module provides services for the management of all aspects of the user account, user preferences and authorization control.
All the logic of the module resides at the BLS side that also acts as a server for the user management services. The front end of the module is a part of the application frontend and uses web services in order to interact with the server.
It should be emphasized that the UM module does not deal with the actual implementation of the various definitions over the components of the application. This will ultimately have to be done independently by the components. For instance, the module may be used to define an authorization policy for a group of users, but the actual implementation of that policy over the data that is stored in the MOSAICA database will have to be carried out by the components that deal with that data.

4.1 Layers

- The authorization module consists of 2 layers:
  - User Interface
  - Server side
- As a general rule we aimed to implement as much functionality as possible on the server side and render the user interface an interactive part that user services whenever in has to execute any logic.
- The communication between the two layers is implemented through Web Services of type 'Document' (WSDL Binding style).



4.2 Entities

The following entities exist in the UM module:

- **User** – A single account that holds MOSAICA user's details

- **Interest Group** – A group of Users that are interested in the same range of knowledge and information of what MOSAICA has to offer.
- **Authorization Group** – A group of Users that have the same access rights to MOSAICA modules and MOSAICA knowledge base.
- **Preferences** – A set of preferences that a user holds, that apply to the wat this user wants to run or view the application

4.2.1 User

User accounts hold all the details of MOSAICA users. Every user account is linked to an authorization group that enforces the authorization policy for that user.
Basically, there are 3 types of user accounts:

1. Ordinary adult account
2. Student account
3. Child account

Student accounts are opened for students of schools or any academic institute that participate in the MOSAICA project.

- A student account will always be subjected to a super account that will be held by a tutor/teacher/other authority and that will impose authorization policy over that student account.
- Every editing action (i.e. creation of a Virtual Expedition, additions to the knowledgebase, semantic annotation, commenting of Virtual Expeditions, uploading of private content) by student requires authorization of the user group moderator. Such modification will be immediately visible only to the student and the user group moderator, and only after the authorization also to others. During authorization all information regarding the editing can be overridden by the user group moderator.

Child account: It has been agreed that children under the age of 13 will not be able to independently register into MOSAICA. It is of course impossible to fully enforce such decision, but we do act upon the assumption that people who register into MOSAICA give their correct personal detail.
In the light of that decision we have created a special child accounts that are restricted in actions and subject to parent accounts.

- In order to enable a child under the age of 13 to use MOSAICA, a special parental account must be opened. This account is opened by the parent or the legal guardian of the child in child's name. The parent may subsequently open any number of child accounts under his own account.
  The parent will also be able to farther update the child accounts, which the children will not be allowed to do.
- Every editing action (i.e. creation of a Virtual Expedition, additions to the knowledgebase, semantic annotation, commenting of Virtual Expeditions, uploading of private content) by a child requires authorization of the parent. The modification will be immediately visible only to the child and the parent, and

only upon the parental authorization also to others. During authorization all information regarding the editing can be overridden by the parent.

The following characteristics apply to all types of user accounts:

- Every user account includes the following properties (**(M)** = Mandatory):
  - **Status** – Active / Inactive **(M)**
  - **Type** – Child/Parent/Moderator/Student/Regular **(M)**
  - **User ID (M)**
  - **Password** – Encrypted **(M)**
  - **User Name** – Used primarily for display purposes. The name that is displayed on the screen when the user has logged in, and the name that the other users will see when they wish to refer to the user. **(M)**
  - **Super User ID** – The ID of a user account that controls current account.
  - **Interest Group ID** – The ID of the interest group that the user belongs to. This property may hold multiple values. **(M)**
  - **Authorization Group ID** - The ID of the authorization group that the user belongs to. **(M)**
  - **Email** - The email is never disclosed to anybody and is used for identification and authentication of the user **(M)**
  - **Gender** - By default this information is not disclosed. This information is used solely for anonymous statistics required for evaluation of MOSAICA. Child accounts of all types are not required to specify gender.
  - **Display Gender** – Controls whether Gender will be displayed in places of public access
  - **Birth Date -** By default this information is not disclosed. This information is used solely for anonymous statistics required for evaluation of MOSAICA. Child accounts of all types are not required to specify gender.
  - **Display Age**– Controls whether Age will be displayed in places of public access
  - **Country -** By default this information is not disclosed. This information is used solely for anonymous statistics required for evaluation of MOSAICA. Child accounts of all types are not required to specify gender.
  - **Display Country**– Controls whether Country will be displayed in places of public access
  - **ID Question** – A question that the user will be asked in case he has forgotten his user ID and/or Password (for instance: *What is the name of your pat?).* **(M)**
  - **ID Answer** – An answer to the above question. **(M)**

- Every account may or may not be linked to a super account. If it does, then it will inherit all the group links from the super account and it will be managed solely by the super account (which means that it will be opened, updated and closed by the super account).

- If an account has a super account, then its authorization may only be equal or smaller then the super account (i.e. the set of authorized objects will be a subset of the one of the super).

4.2.2 Interest Group

Interest groups are groups of users that are interested in the same domain of knowledge of what MOSAICA has to offer.
Their significance is primarily the narrowing of the amount of data the a user has to cope with. Since it is expected that the MOSAICA knowledge base will grow rapidly to incorporate millions of items, we wanted to provide the user with means to simplify that labyrinth of knowledge and let himself focus on the area of his interest. The interest groups are the tool that facilitate for it.

- Every interest group will include the following properties (**(M)** = Mandatory):
  - **Status** – Active / Inactive **(M)**
  - **Group ID** **(M)**
  - **Group Name** **(M)**
  - **Group Moderator** – This attribute may hold multiple values **(M)**

- An interest group denotes an area of common interest for a group of users.

- When a user is linked to a group of interest, he is allowed to view and further contribute to materials that have been entered by members of the same group, even if the materials have not yet been published (approved by the board of directors).

- A user may belong to any number of interest groups.

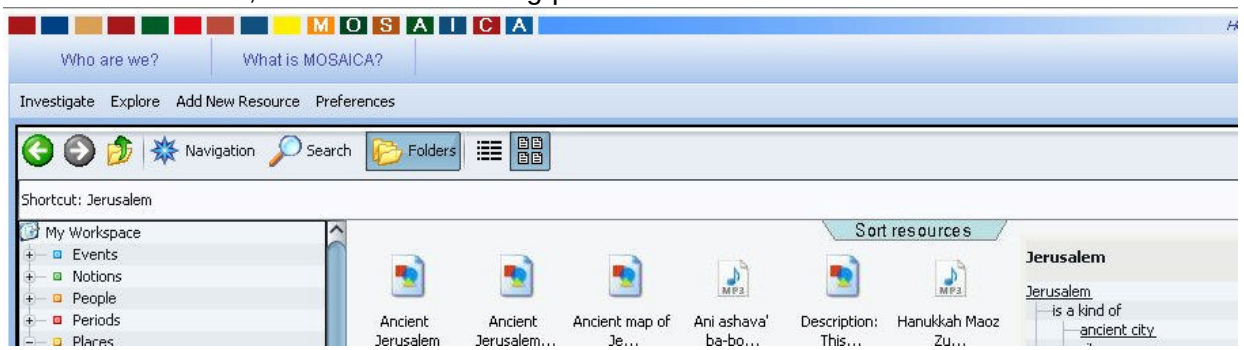
4.2.3 Authorization Group

An authorization group is a set of access an usage rights that is granted to a group of users. The Authorization groups are set in an hierarchical order, where the groups on top will always have more rights than the groups below them in the hierarchy. Any user of MOSAICA must be bound to an authorization group from which he derived all the access and usage rights.

- Every authorization group will include the following properties (**(M)** = Mandatory):
  - **Status** – Active / Inactive **(M)**
  - **Group ID** **(M)**
  - **Group Name** **(M)**
  - **Group Moderator** – May hold multiple values **(M)**
  - **Parent Group** – A link to the parent group **(M)**

- An authorization group is an entity that defines usage rights for various action object of the application.

- An action object is any object on the screen that invokes some functionality when the user clicks on it.
  For instance, consider the following picture:



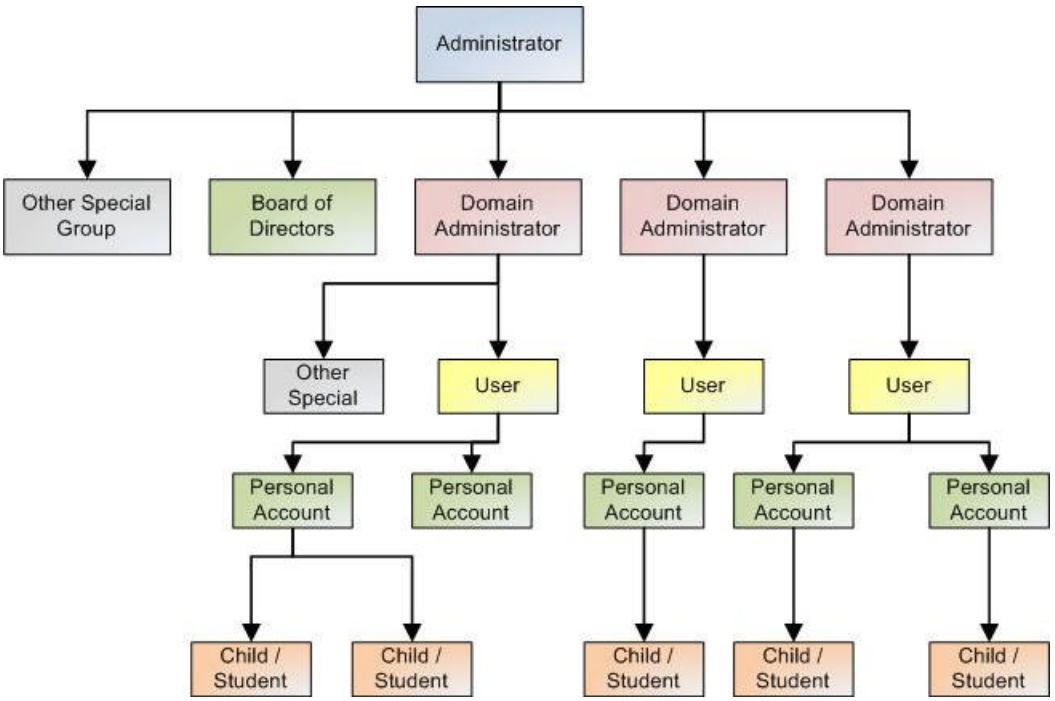
  Any of the menu items can be considered as an action object, so for instance, it can be defined that for a certain authorization group the 'Add New Resource' link is disabled.

  Similarly, this mechanism can be used in order to limit access to elements within the various components.

- How to apply the authorization is in the hands of the people who actually write the various components. The authorization module doesn't actually "know" how the action items for which an authorization is defied are being used and will only provide web services, which when given a user ID, will return a list of all the authorized action items for this user.

- An authorization group must inherit from another authorization group. Basically, it inherits the list of authorized action items from its parent group.
  A group who is a child of another group, will be allowed to have only a subset of the parent group action items, which means that the child can have action items disabled when they are enabled for the parent but not the other way around. Consequently, the group on top of the authorization group hierarchy will have the largest number of action items authorized for it and the descendant groups will have less and less authorized items the more they will be down the hierarchical structure.

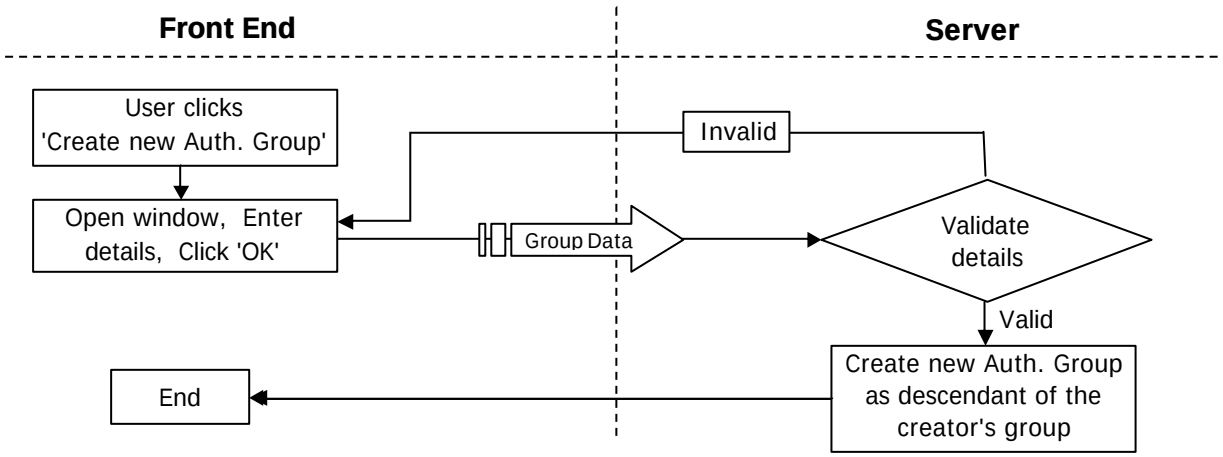
- Only one authorization group does not inherit from any other group, the System Administrator group for which all action items are enabled by default.
  This group stands on top of the group hierarchy and all the other groups are descendants of that one.
  Here is a diagram that shows the basic hierarchy of the authorization groups:



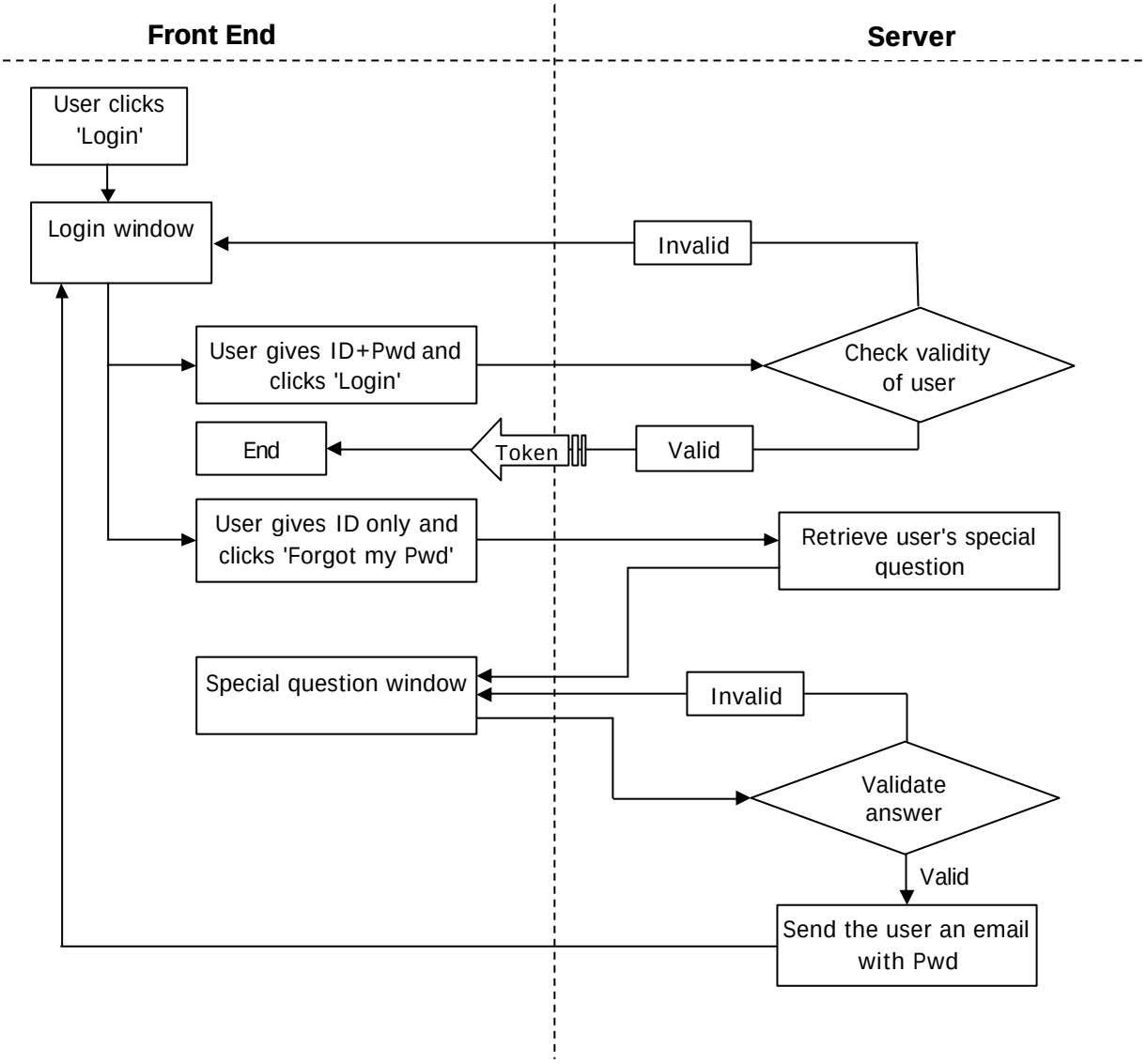
4.3 Workflows

Here are some sample workflows that are implemented within the user management module:

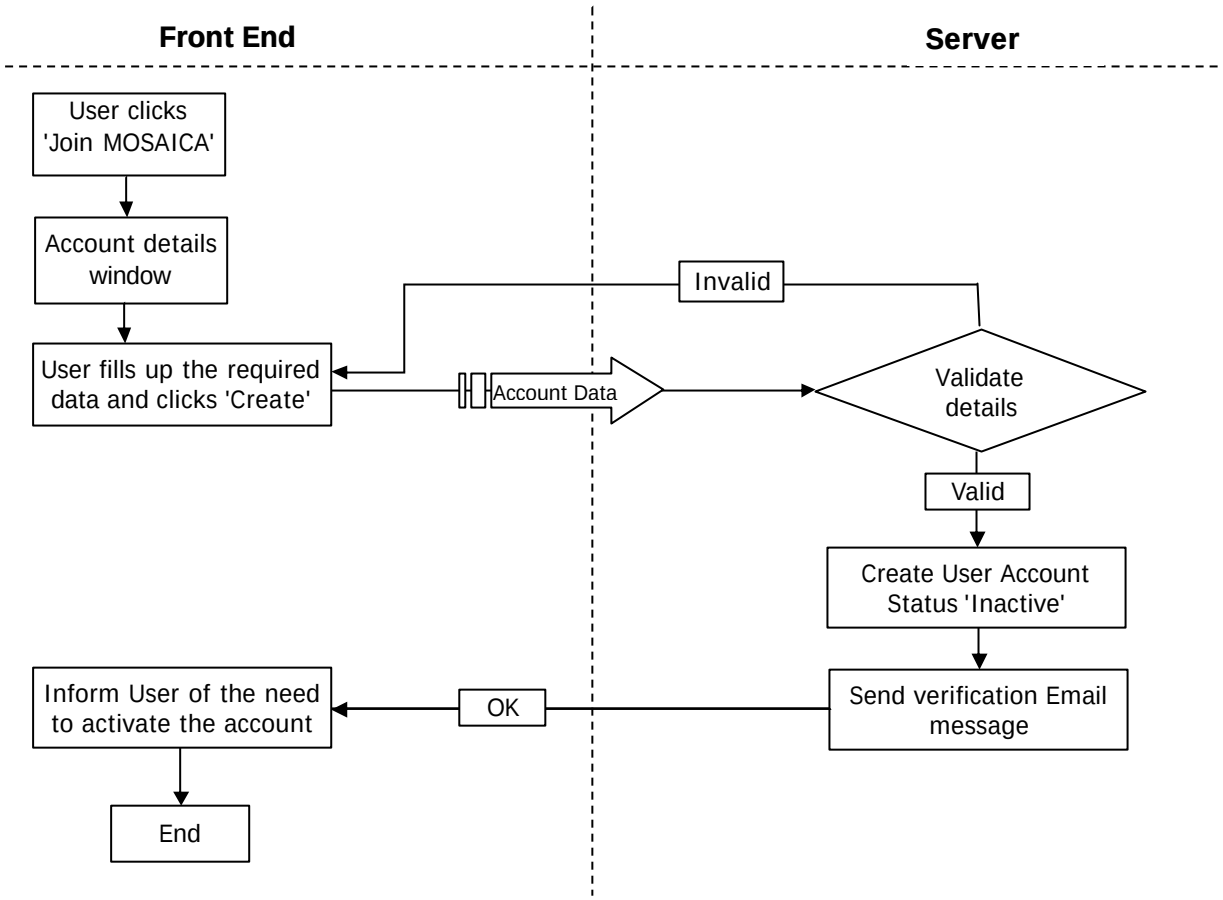
4.3.1 Create new authorization group



4.3.2 Login to the system



4.3.3 Open new user account



5 ONLINE COMMUNICATION

The MOSAICA application also provides an online communication utility as an additional layer for the MOSAICA users to communicate between themselves.
Once a user is logged on into the MOSAICA application, he is automatically added to the 'Online Users list' and his name appears automatically on the screen.
The application provides a special 'Slide in Slide out' frame within which all the names of all the online users appear.
By clicking on a user's name, you can chat with him online.
This feature is there to help MOSAICA user groups to interact between themselves and there for be more productive when contributing to MOSAICA and also when searching over the knowledge base.

6 APPENDIX: LIST OF 3RD PARTY TOOLS

Front end:

- *Google Web Toolkit (GWT)* - serves as the basis for developing the graphical user interface (GUI). This provides the ease of developing the graphical user interface in Java, while at the same time results in a streamlined and responsive web interface.
- *Apache/Tomcat* - serves the web application for the users' web browsers.
- *Axis (and REST WS)* - Axis framework is used for calling the web services (WS). More importantly, the Axis and our implementation of the REST WS is used for coordination of the underlying distributed components, which are used in MOSAICA.

Back end:

- *NetBeans* – Used in order to create EJB and servlets to run on the Glassfish application server.
- *Glassfish* – An open source application server that is part of the GNU project. Used as a platform to run all the JAVA components which provide the services to the application.
- *BPEL* – Used as a platform for the creation and running of workflows.