CENG 491
SENIOR PROJECT

Initial Design Report

Macro-Hard

PROJECT TITLE: Fizan

COMPANY STAFF:	Uğur Altun	1448356
	Kürşat Aksakallı	1448315
	Kemal Beşkardeşler	1448455
	Ömer Faruk Çelik	1448547

Index

1. INTRODUCTION

Purpose of the document:

The purpose of this document is to show initial design of our project “FIZAN”, and introduce our project with all needed diagrams for you to learn about the project and for us to plan our works we will do in the future.

2. PROJECT DESCRIPTION

2.1 Problem Definition

Nowadays usage of mobile phones is widespread and almost everybody use mobile phones in daily life. Therefore mobile phones need to provide most of the required features a person may need in daily life. It is needed to make mobile phones something that makes people life easier in daily life. We can consider such situations:

When a person wants to meet with a friend in some place at some time, it is difficult to arrange this. Many calls or many message sending is needed, for this simple thing. If it is a meeting with many people it is harder and more expensive. Therefore we can imagine such a feature that makes a place visual to all mobile phones. Then we can put an event on this map which can be seen from our friends, which has meeting time and exact position on the map will be useful to arrange these kinds of meetings.

For parents watching their children is a big matter. We can add a feature to mobile phones which permits parents to see their children position on a map, and it will give a warning when children go somewhere else parents don’t want.

A person may need to learn some information around such as what is where, where to find a good shopping center, a cinema, etc. With the developing technologies after now, providing environment knowledge of a person is a requirement. These information may be about advertisements and campaigns of shopping places and social places, and other places where a person may need such as hospitals, police station, taxi rank, etc.

With seeing these problems we conclude such a feature is essential for mobile phones, that is seeing other people on a real world map, and communicate with other people by using this map featured environment.

2.2 Project Features

Our project will answer the problem we have defined previously. FIZAN will provide a map based environment to mobile phones. This is not done in our country and people need such a product after the developed technologies on mobile phones. Since 3G is coming to our country we will be able to use these features after now. 3G is the third generation wireless phone technologies which makes internet for mobile phones faster. And this will enable us to produce more developed products which need fast internet connection. After 3G we can do our project but it is not a project based on 3G, we know 4G is on the way and more developed technologies will come, that means, our project will become more useful in the future.

We will do our project for Android OS based mobile phones. Why we have chosen Android is that, Android is a Google product and we think it will be more popular and will be used most of the mobile phones in the future. Since we want our product to be used for a long time Android will be good for us. Also developing programs in Android is simple, and it enables to use Java as programming language. And also some features of Android makes it better for us. First of all it is free and easy to use with Eclipse as a Java ME. Android provide good interfaces and enables to use Google products such as GTalk, Google Maps, etc. Beside all of these, since it is a new technology and a Google product, we think it will be widespread and a product done with Android will be valuable.

A project with all features we described has not been done yet. Especially parent mode which enables parents to watch their children by phone will be a wonderful for people and will heal the society's a big problem. Parent will be able to protect their children by this way from bad situations and prevent them to go places harmful for them. Also users will find creating meetings very useful and helpful. They will use this feature in daily life very much. We think our project will be an enjoyable product for people and will give a good taste on their social life.

FIZAN is a good social program which will enable people to access Facebook at any place. And also when a user takes a photo and wants to add it to his/her account on Facebook then he/she does not need to wait to put it, because with our program immediately he/she can add photo to Facebook. Also it will be powerful as MSN we think because user will be able to chat via mobile phone. Seeing friends on the map and according to their situation beginning to talk at any where will be more inviting than just sitting on a computer and chatting. We hope people like our FIZAN.

Project will have some main features:

Visualizing: We will show the real world on a map. We are thinking to take the map from Google Maps.

Showing friends: People which user has the permission to see will be visualized on the map by taking their location information (It will be taken from GSM company or

phone cells GPS). When user chooses a person several actions will be ready to be chosen.

Instant messaging: User will be able to start instantly messaging with another user. We are thinking to do this by GTalk.

Giving advertisement: Some social places and shopping centers will give advertisements and information about their campaigns which will be shown on the map.

Modes: There will be modes for users to switch which will increase the functionality and range of people that product able to reach.

Client & server side: Program will communicate with database and other mobile phones via this feature.

Social Networks: Product will communicate the social networks such as Facebook. User can directly communicate with these kinds of services, send and get information.

Web part: Almost all of the features that cell-phones side has in the project will be added to web.

2.3 Design Goals & Objectives

2.3.1 Usability

Our project will be very easy to use because it is a mobile phone technology which we want all people with a mobile phone can use this product. Therefore it has to have a usage which is very clear for everyone. Also we will provide some modes which will enable or disable some features of the product that will make usage simpler for those who cannot use all features.

2.3.2 Security and Privacy

Since if everybody sees all other people on the map and all other people's information such as where he/she is at the moment, it will be bad for everyone. Therefore we will protect users' information from other users although that user gives some permission to see some kind of information. That is a user can see another user if only other user has given permission that user to see him/her. Also user will be able to hide himself/herself on the map at any time; therefore our product secures users' private life in daily life. However we think to give permission parents to see their children at any time, but it may be optional too.

2.3.3 Availability

Our product will be easy to get. People will be able to get it from the internet by downloading to the personal mobile phone, that simple.

2.4 Design Problems

2.4.1 Time Problems

Our project's finish time is determined by ceng490 syllabus. Detailed schedule is given in the schedule part.

2.4.2 Software Constraints & Requirements

We will write our project mainly in Java, to the mobile phones with Android OS, using Eclipse. These are our main software requirements:

Web Server (JSP): It is Java technology that allows software developers to dynamically generate HTML, XML or other types of documents in response to a Web client request.

DBMS (MySQL): MySQL is a powerful database management system which we can be used to manage and store our data in the server side.

Eclipse: It is an open source IDE which makes possible and easier to develop codes for Android.

Android OS: An operating system for mobile phones which enables to produce programs for mobile phones.

GTalk: It is a Google product for instant messaging and voice over internet protocol (VOIP).

3. DATA DESIGN

In the application Users, Friend lists, Events are hold in database. Due to the difficulties in establishing database in Client side (Web environment or cell-phone environment) and retrieving data from database in Client side, database will be established in Server side.

In this section we will look at the database tables and ER (entity relationship) diagrams.

3.1 DATABASE TABLES

Users:

In server side, all the users and user's data will be hold in the database in the name of "Users" table. This table includes all the information that belongs to user.

- **User_id:** All users have unique id,
- **User_name:** Name of the user**,** has default value " ".
- **User_passwd:** Password for the user, has default value "1234"
- **User_gmail_account name:** User's gmail account
- **User_facebook_mail_address:** mail address used in Facebook.
- **User_tel_num:** telephone number of the user
- **User_address:** Open address of the user
- **User_country:** Name of the Country
- **User_city:** Name of the city
- **User_parents_tel_no:** Phone number of parents which can be used in parent Mode
- **User_log_file_name:** Name of the log file which track of the user kept
- **User_border_point1_x:** Latitude of first point which will be used in defining borders of the parent mode.
- **User_border_point1_y:** Altitude of first point which will be used in defining borders of the parent mode.
- **User_border_point2_x:** Latitude of second point which will be used in defining borders of the parent mode.
- **User_border_point2_y:** Altitude of second point which will be used in defining borders of the parent mode.
- **User_avatar:** File name of the avatar picture.
- **User_last_online_time:** Date of the last online time

User_id	INTEGER
User_name	STRING
User_passwd	VARCHAR(20)
User_gmail_account name	VARCHAR(100)
User_facebook_mail_a dddress	VARCHAR(100)
User_tel_num	VARCHAR(20)
User_address	VARCHAR(500)
User_country	VARCHAR(30)
User_city	VARCHAR(30)
User_parents_tel_no	VARCHAR(20)
User_border_point1_x	INTEGER
User_border_point1_y	INTEGER
User_border_point2_x	INTEGER
User_border_point2_y	INTEGER
User_avatar	VARCHAR(50)
User_last_online_time	Date
User_log_file_name	VARCHAR(50)
PrimaryKey(User_id)	
Constraint NOT NULL (User_passwd), NOT NULL (User_tel_num)	
NOT NULL (User_name)	

Friends:

Friend list will be hold in this table. For each of the users all of the friends will be kept here.

- **Fri_id1:** Who has the friends
- **Fri_id2:** User with id1's friend
- **Fri_type:** `How user whom has id1 wants to appear to user with id2. (1 for Online, 2 for busy, 3 for offline)
- **Fri_dist:** What is the alert distance for id1 and id2
- **Fri_online:** Is id2 online ?

Fri_id1	INTEGER
Fri_id2	INTEGER
Fri_type	INTEGER
Fri_dist	INTEGER
Fri_online	INTEGER
PrimaryKey(Fri_id1, Fri_id2)	
ForeignKey(Fri_id1), ForeignKey(Fri_id2) references Users	

Events:

This table holds the detailed data for events. Therefore events can be specified and classified.

- **event_id:** Unique id of the event
- **event_cat:** Category of the event (i.e. 1 for entartainment, 2 for shopping…)
- **event_start:** Day of starting
- **event_finish:** Day of finishing
- **event_time:** Time of the event if any (i.e. Concert time)
- **event_fare:** Cost of the event if any (i.e. Concert fee)
- **event_owner:** Name of the owner of the event (maybe company name)
- **event_loc1:** Latitude of the location
- **event_loc2:** Altitude of the location
- **event_region:** Local name of the event (i.e. Kizilay)
- **event_city:** Name of the City
- **event_off_addr:** Formal address
- **event_unoff_addr:** Informal address
- **event_tel:** phone number related to event
- **event_link:** Web link for detailed info
- **event_desc:** General info for event or slogan for advertisement
- **event_foto_name:** Name of the photo which will be displayed
- **event_view_type:** numbered type of the event view (like 1 for just photo or 2 for photo with description)

event_id	INTEGER
event_cat	INTEGER
event_start	Date
event_finish	Date

event_time	Date
event_fare	INTEGER
event_owner	VARCHAR(100)
event_loc1	INTEGER
event_loc2	INTEGER
event_region	VARCHAR(50)
event_city	VARCHAR(50)
event_off_addr	VARCHAR(300)
event_unoff_addr	VARCHAR(300)
event_tel	VARCHAR(20)
event_link	VARCHAR(100)
event_desc	VARCHAR(300)
event_foto_name	VARCHAR(30)
event_view_type	INTEGER
PrimaryKey(event_id)	
Constraints NOTNULL (event_cat), NOTNULL(event_start),	
NOTNULL(event_finish), NOTNULL(event_loc1), NOT NULL(event_loc2)	

User-Event Settings:

This table holds the setting information for user about events. User-Event Settings table can be combined with Users table but for the performance issues it is separated.

Setting_user_id: Id of the user
Setting_event_type: Category of the wanted event
Setting_event_dist: Maximum distance of the event.
Setting_event_interv: Time gap between re-controlling the event. (in minutes)

Setting_user_id	INTEGER
Setting_event_type	INTEGER
Setting_event_dist	INTEGER
Setting_event_interv	INTEGER
PrimaryKey(user_id, even_type)	
ForeignKey(user_id) references Users	

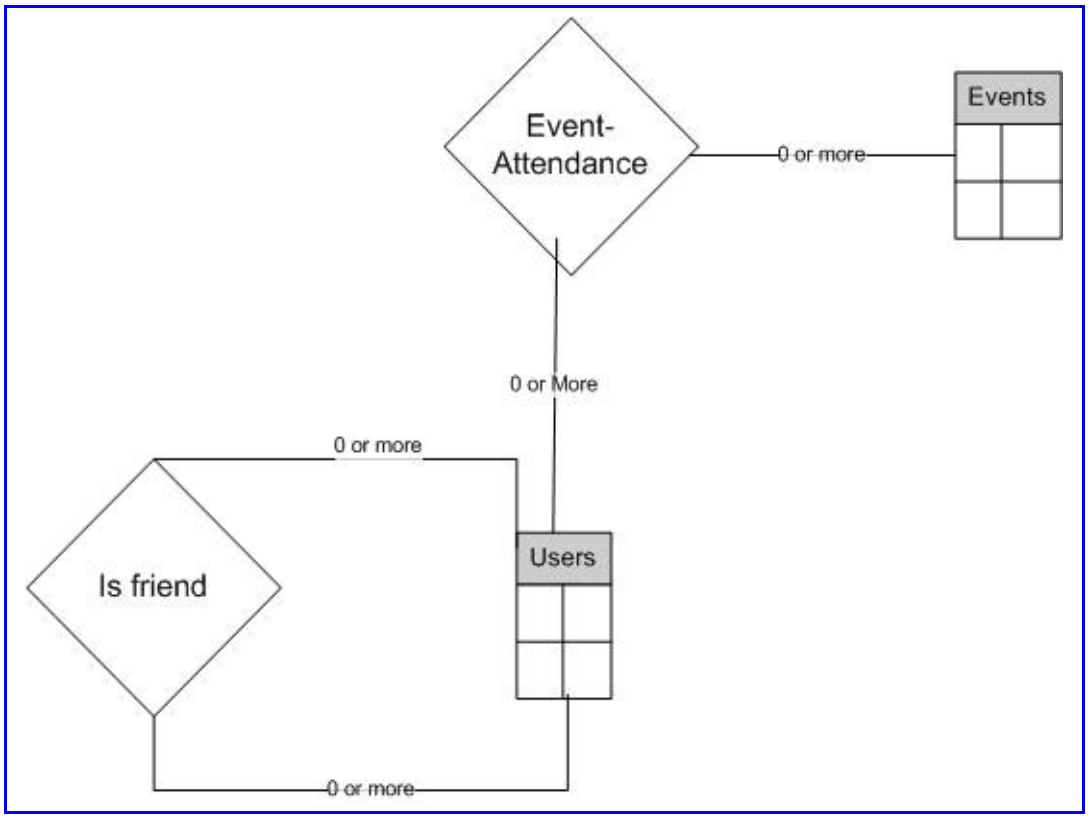
Event-Attendance:

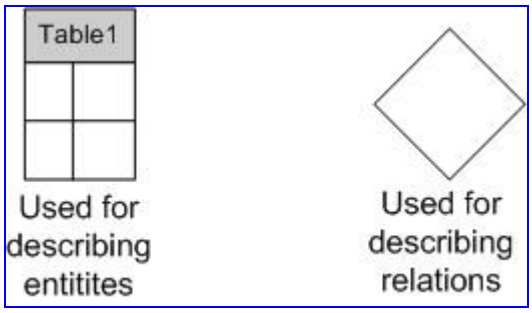
In this table user's events are kept and also attendance information.

- **User_id:** Id of the user
- **Event_id:** Id of the Event
- **Attend_status:** Attending status (1 for Yes, 2 for Maybe, 3 for No)

User_id	**INTEGER**
Event_id	**INTEGER**
Attend_status	**INTEGER**
PrimaryKey(User_id, Event_id)	
ForeignKey(User_id) references Users, ForeignKey(Event_id) refererences Events	

3.2 ER DIAGRAMS





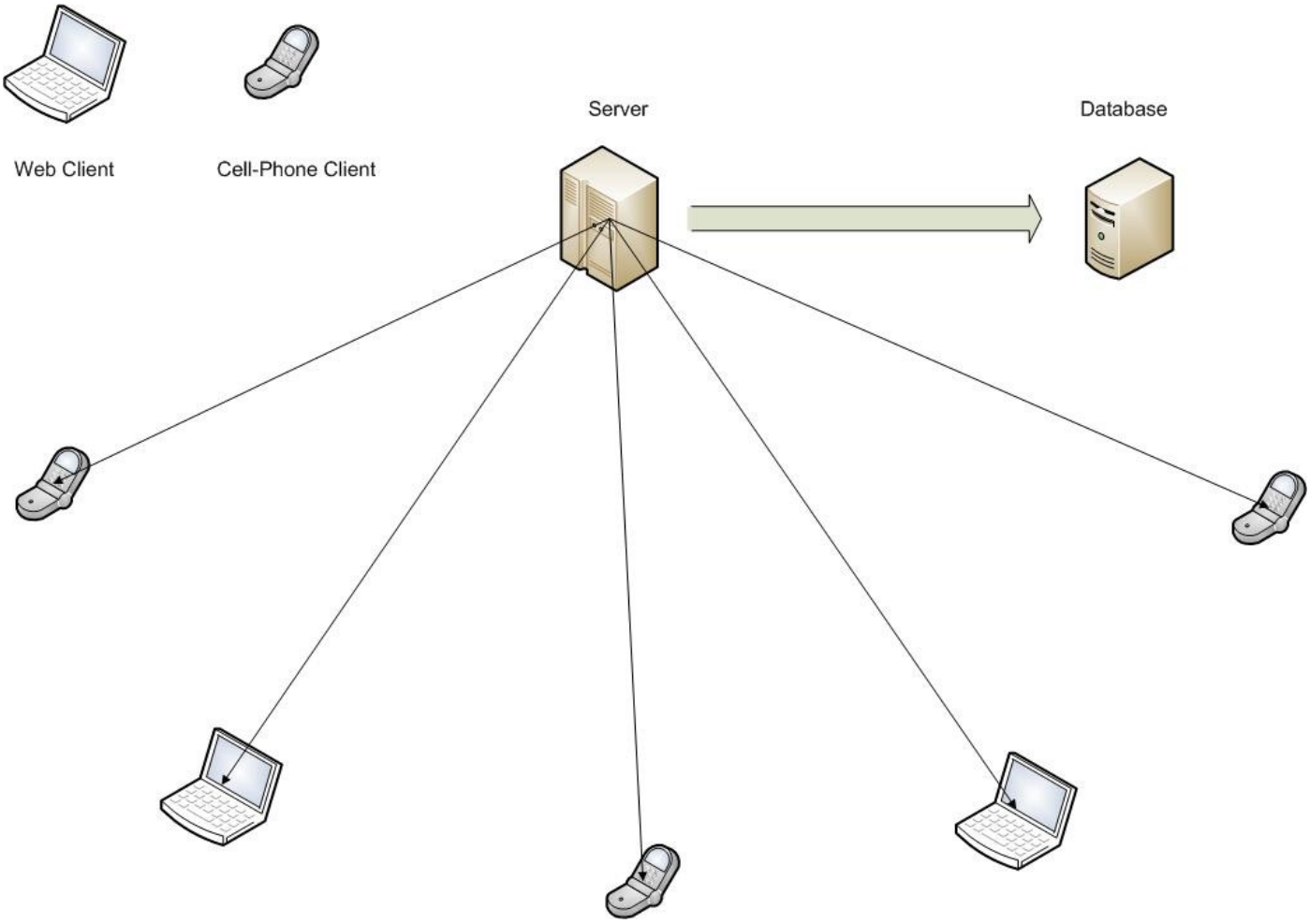
In The ER (Entity relationship) diagram above, there are two entities and 2 relations. First relation is “Is Friend” relation. This shows the relation between two users. And give the answer to the question “is users are friends or not”.

As can be seen clearly, second one shows the relation between users and events and gives idea about whether user attends the event or not.

4. ARCHITECTURAL DESIGN

In the project “Fizan”, the modules are totally independent of each other; basically there are two modules in project.

1. Server side
2. Client Side
   2.1 Cell-Phone Client
   2.2 Web Client

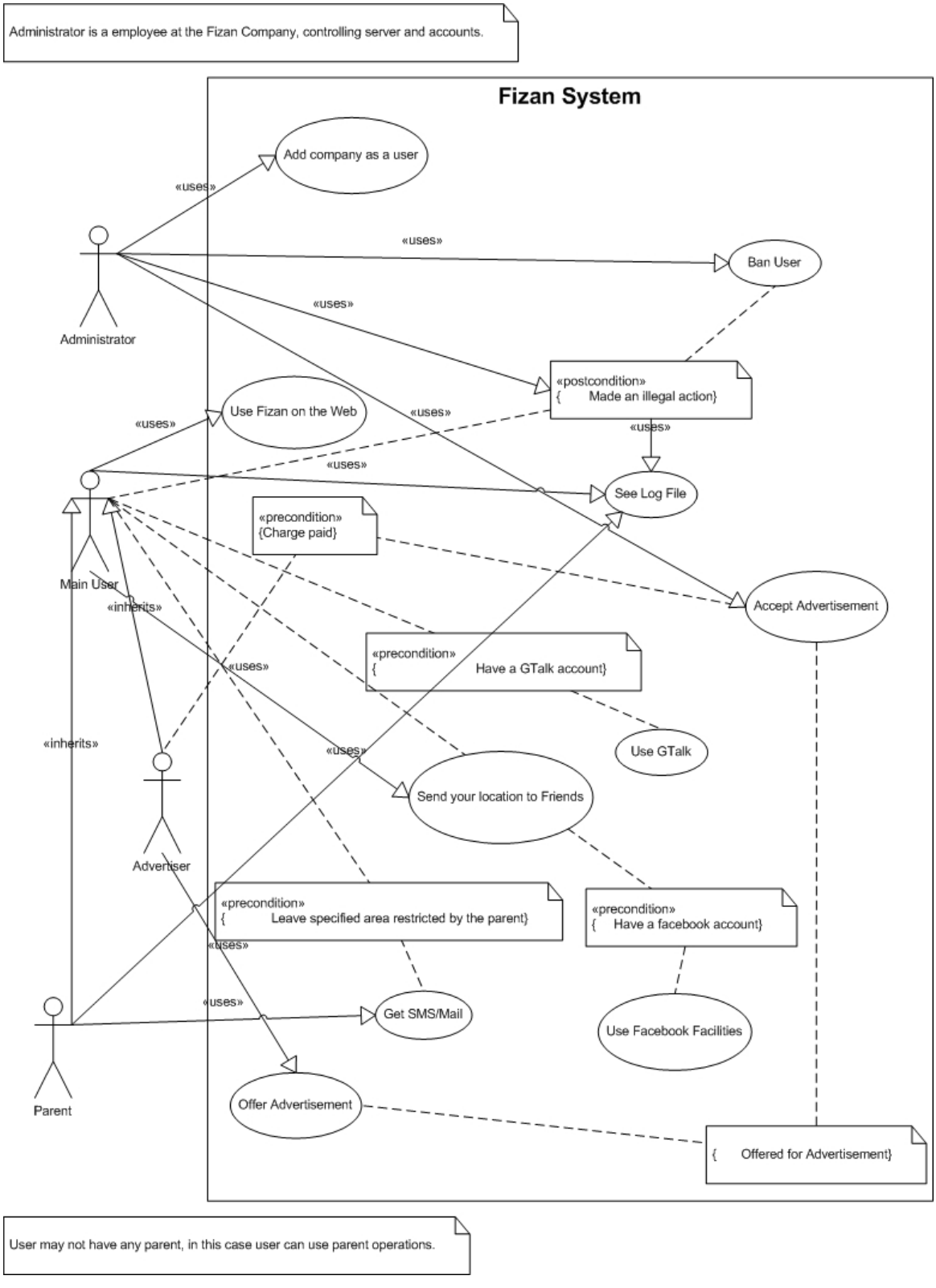


The current idea is, server does almost all the operations, fetching data from database, updating database, doing necessary calculation, getting data from clients and sending data to clients.

For the client side, Client sends request to the server and gets data from the server. So, Client side is not bothered with calculations. Generally, client shows the data coming from server on its user interface. For example, client sends friend list to server and wants to take location data from friends once an hour.

5. SYSTEM DESIGN

5.1. Use Case Diagram

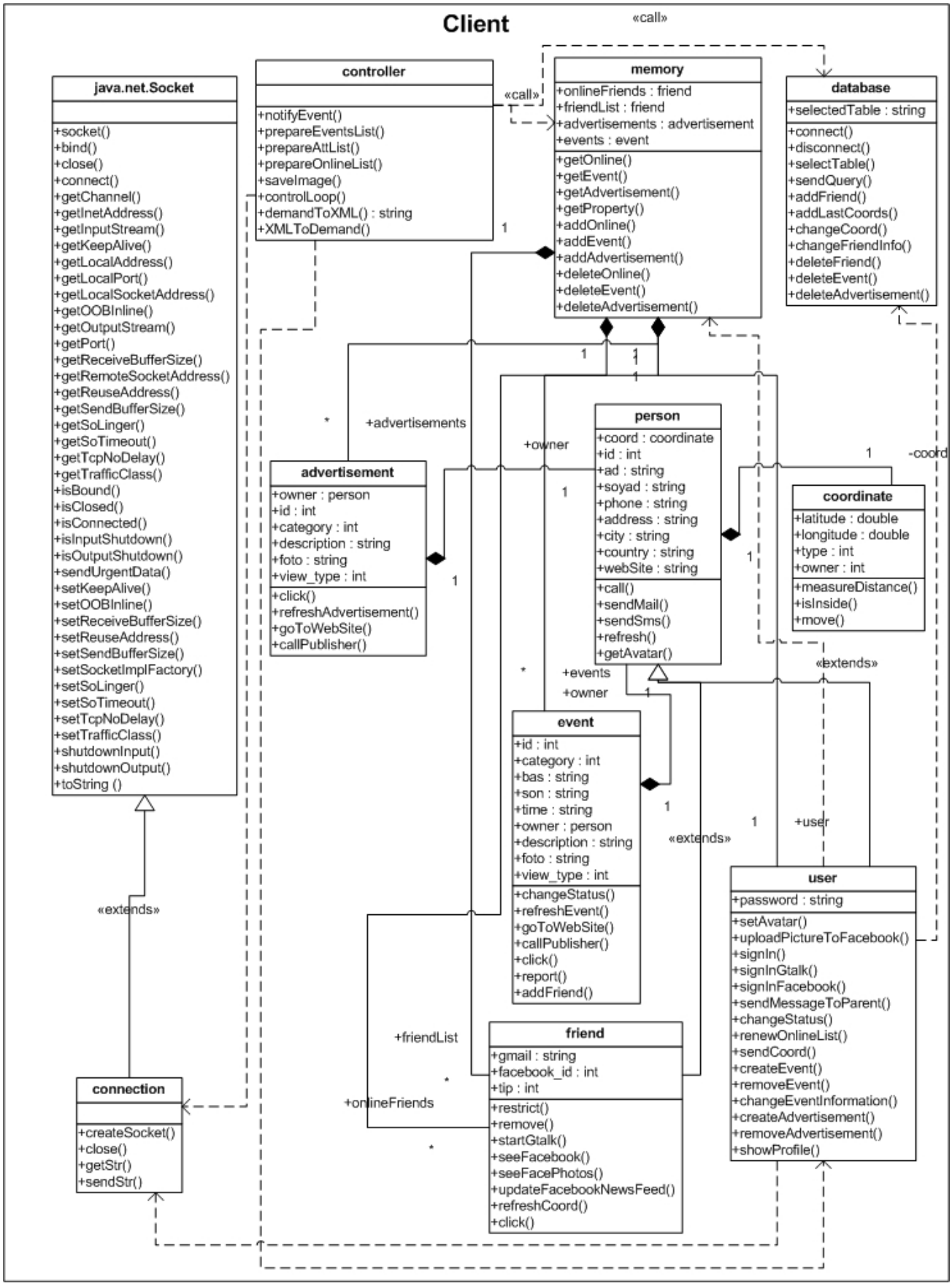


5.2 Classes and Class Diagram

Roughcast of the Classes and the Work Design

Fizan's code is divided into two, some to the server side and some to the client side. At the server side, majority of the process is database processes. At the server, there are some tables about situation which is being updated. These tables keep online people, informations about people, friend linked list, events, advertisements, settings, event attendance, photo informations, coordinates. When any of these has changed, client send information to the server and updated here. When there is any demand from clients, server sends that information. These demands will be periodically or manually. Client program takes data from server and inform the user. Briefly, data of the users are in the server database, and the client program takes other data related to it from the server and after design, serves to user. Controller classes at the both side works and uses other classes. Pool's main loop method control aal of the things not belong to the clients. At server side, for every connection with the clients, there is a contoller class, their controlLoop works in threads.

5.2.1 Client Class Diagram



Client Classes:

connection: This class communicates with the server side communication class. When any demand occurs in the program activated by the user, this class gets instruction from other client classes, and sends information to the server. This class serves as an interface to the client classes. This class listens server's connection class to take new informations immediately.

Methods:

createSocket: Creates socket.
close: Closes socket
getStr: Gets string from other side.
sendStr: Sends data to other side.

controller: Understands what is received from server and makes some file operations and provides display.

Methods:

notifyEvent: Notifies the user about an event of his/her friend and asks to read/ignore.
prepareEventsList: Prepares events/advertisements list that the server responded to user's demand.
prepareAttList: Prepares event.friendsAttList/friendsMaybeAttList.
prepareOnlineList: Prepares online friends list.
saveImage: Saves received image to a directory/file.
controlLoop: This method is available when fizan is processing in the phone. It waits for responses coming from the user or server, when it comes it uses its own classes methods' to control requests.
demandToXML: Converts demands to XML strings.
XMLToDemand: Converts XML strings to objects.

person: It keeps general personal information. id, name, avatar, phone number, address, mail address, web site, coordinate, are data kept in this class.

Methods:

call: Calls the person.
sendMail: Sends mail to the person.
sendSms: Sends sms to the person.
refresh: Takes new information about friend's account from server.
getAvatar: Takes avatar picture of the person.

friend: This class extends person class. It has some additional data about friends and additional methods. Gmail(keeps gmail account),facebook.

Methods:

restrict: Change status and restrict friend.
remove: remove friend from list.
startGtalk: Starts a conversation between user and friend using gtalk.
seeFacebook: open web gadget to see friend's facebook page.

seeFacePhotos: Go to the web page, which shows photos of the friend.
updateFacebookNewsFeed: Gets unread news of friend from facebook.
refreshCoord: Gets new coordinate values of the friend.
click: Displays information when friend clicked on the screen.

user: This class extends person class.

Methods:

setAvatar: Sends avatar picture to the server to set avatar.
uploadPictureToFacebook: Uploads a picture to specific facebook album.
signIn: Sign in to fizan.
signInGtalk: Sign in to the Gtalk.
signInFacebook: Sign in to the facebook and goto facebook home page.
sendMessageToParent: Sends a message to the parent.
changeStatus: Change user status and sends information to the server.
renewOnlineList: Renew online list.
sendCoord: Sends current coordinate to the server.
createEvent: Displays a window, in which you can adjust event's information, select friends to be notified,etc. And after completing final processes, necessary operations will be completed on the phone and on the server.
removeEvent: Remove any created event. (as create event)
changeEventInformation: Opens event's information dialog to be changed by the user.
createAdvertisement: Create an advertisement.
removeAdvertisement: Remove any created advertisement.
changeAdvertisementInformation: Opens advertisement's information dialog to be changed by the user.
showProfile: Displays the user's own profile.

event: This class keeps event informations. friendsAttList(attending event), friendsMaybeAttList(maybe attending event)

Methods:

changeStatus: Changes attendance status of the user for this event.
refreshEvent: Gets event information.
goToWebSite: If exists, goto web site.
callPublisher: Call event's owner.
click: Displays event information on the screen.
report: Prepares report about the event and sends it to the admins.
addFriend: Adds friend to the list of friends who will attend the event (discrete list is used for people whose attending status is set as "maybe").

advertisement: This class keeps information about advertisements and when client want to make any process about advertisement, this class will be used.

Methods:

click: Displays advertisement information on the screen.
refreshAdvertisement: Gets advertisement informations.
goToWebSite: If exists, goto web site.
callPublisher: Call advertisement's owner.

coordinate: Keeps information of the coordinates. Latitude and longitude are kept in this class.

Methods:

measureDistance: Measures distance between other objects.
isInside: Is our coordinate inside the rectangle coming as a parameter.
move: Change coordinates referenced to saved coordinates.

memory: This class used to fetch and write back to memory. It arranges memory.

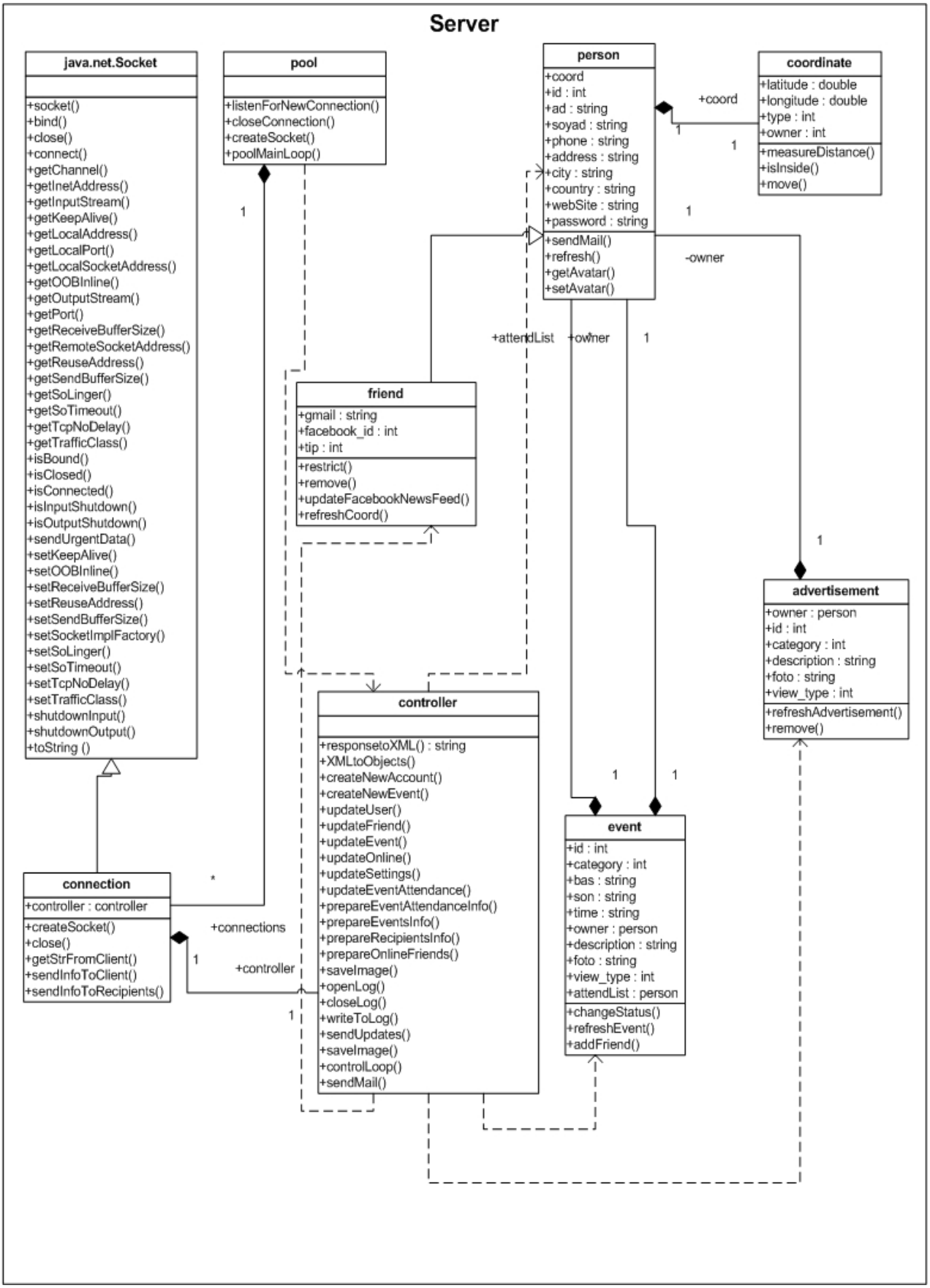
Methods:

getOnline: Get online person from online list.
getEvent: Get an event from vector.
getAdvertisement: Get an advertisement from vector.
getProperty: Gets a specified property from vector, according to parameters.
addOnline: Adds a person to online list.
addEvent: Adds an event to event list.
addAdvertisement: Adds an advertisement to the list.
deleteOnline: Deletes person from online list.
deleteEvent: Deletes event from the list.
deleteAdvertisement: Deletes advertisement from the list.

database: This class used to make database usage easier(In Android, SQLite is our database).

connect: Connect to database.
disconnect: Disconnect from the database.
selectTable: Select a specific table.
sendQuery: Sends specific query which cannot have any method in this class.
addFriend: Adds a friend to friend list.
addLastCoords: Adds a friend coordinate to the table who is recently became online.
changeCoord: Changes a coordinate of a friend.
changeFriendInfo: Changes personal informations of a friend.
deleteFriend: Deletes a friend from the database.
deleteEvent: Deletes an event.
deleteAdvertisement: Deletes an advertisement.

5.2.2 Server Class Diagram



Server Classes:

pool: This class keeps a vector of connection objects and has methods to manage network traffic. It waits a connection, if occurs, create a new socket with it, and add it to the list.

Methods:

listenForNewConnection: Creates a connection object, adds it to the vector of connections. It will be a Java listener for coming connections (to be changed). When new connection created, its controller started to serve in a thread.
closeConnection: If connection between server and client breaks in any way, this method will be called.
createSocket: Creates socket, add it to the vector.
poolMainLoop: Processes instructions about server, not about clients' demands.

connection: This class communicates with the communicate class of the client side. It constructs a network connection as a server with the other side. According to the demands coming from the clients, this class communicates with other server classes.It fetches information from the database or files, and sends desired information to the connection class of the client; or it updates information in tables or files. It serves as an interface to the server classes.

Methods:

createController: Creates a controller object.
sendInfoToClient: Sends prepared information to the client.
sendInfoToRecipients: Informs determined recipients about an event.
close: Closes socket
getStrFromClient: Gets string from other side.

controller: Understands what the client wants and organizes data, received from client to connection object, for database and file operations.

Methods:

responsetoXML: Converts demands to XML strings.
XMLtoObjects: Converts XML strings to objects.
createNewAccount: Creates a new account, assigns a user id, adds user to "Kullanicilar" table.
createNewEvent: Creates a new event/advertisement, assignes an event/advertisement id, adds event/advertisement to "Events" table.
updateUser: Update user information in "Kullanicilar" table.
updateFriend: Update friend information in "Arkadaslar" table.
updateEvent: Update event information in "Events" table.

updateOnline: Update being online/offline information in “Online” table and takes coordinates.
updateSettings: Update settings in “Settings” table.
updateEventAttendance: Update event attendance status in “Event_Attendance” table.
prepareEventAttendanceInfo: Prepares friends list attending an event.
prepareEventsInfo: Prepares filtered events/advertisement package according to the desired properties to be sent to the client.
prepareRecipientsInfo: Prepares event information with recipients declared.
prepareOnlineFriends: Prepares online friends information within declared range.
saveImage: Saves received image to the related directory and assigns a photo id.
openLog: Opens log file to write.
closeLog: Close Log file.
writeToLog: Writes data to the log file.
sendUpdates: Sends updated informations to the user.
saveImage: Saves image to a appropriate directory, names it and change settings for it.
controlLoop: This method is available when this class is in memory. It waits for responses, when it comes it uses its own classes methods’ to control requests.
sendMail: Sends mail.

person: This class keeps only datas about the person, and manages these data.

Methods:

sendMail: Sends mail to specified address using mail server.
refresh: Refreshes personal informations and send it to the client.
getAvatar: Gets address of the avatar picture’s location.
setAvatar: Loads picture and update database and memory for the new avatar information.

friend: This class is child of the person class. It keeps extra data about friends, which user can reach.

Methods:

restrict: Restricts any friend.
remove: Removes some friend from friend list.
updateFacebookNewsFeed: Updates facebook news for the friend, to learn is there any new activity for the user.
refreshCoord: Refreshes coordinate of the friend, and sends it to the client.

event: This class keeps only datas about the event, and manages these data.

Methods:

changeStatus: Changes attendance status for specified user.
refreshEvent: Refreshes event information, and send it to the specified client.
addFriend: After coming any information of attending to this event, sends new friend to the owner of the event.

advertisement: This class keeps only data about the advertisement, and manages these data.

Methods:

refreshAdvertisement: Refreshes advertisement information, and send them to the specified user.
remove: Removes advertisement from the server.

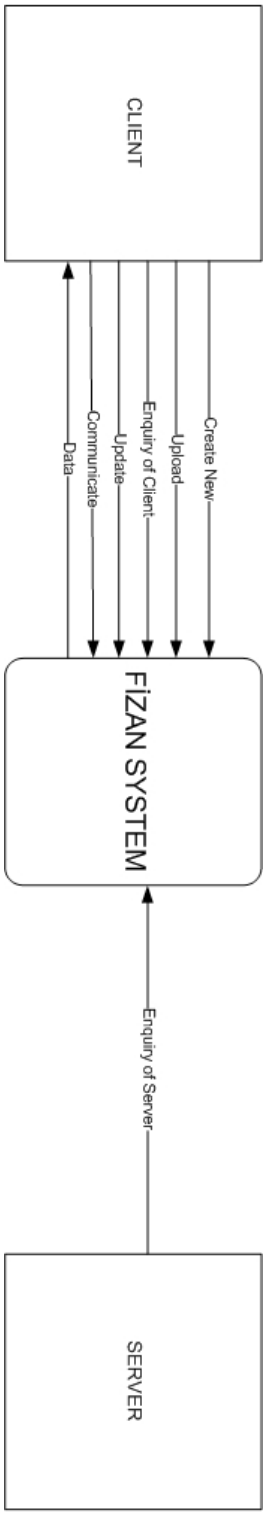
coordinate: This class keeps only datas about the coordinates, and manages these data.

Methods:

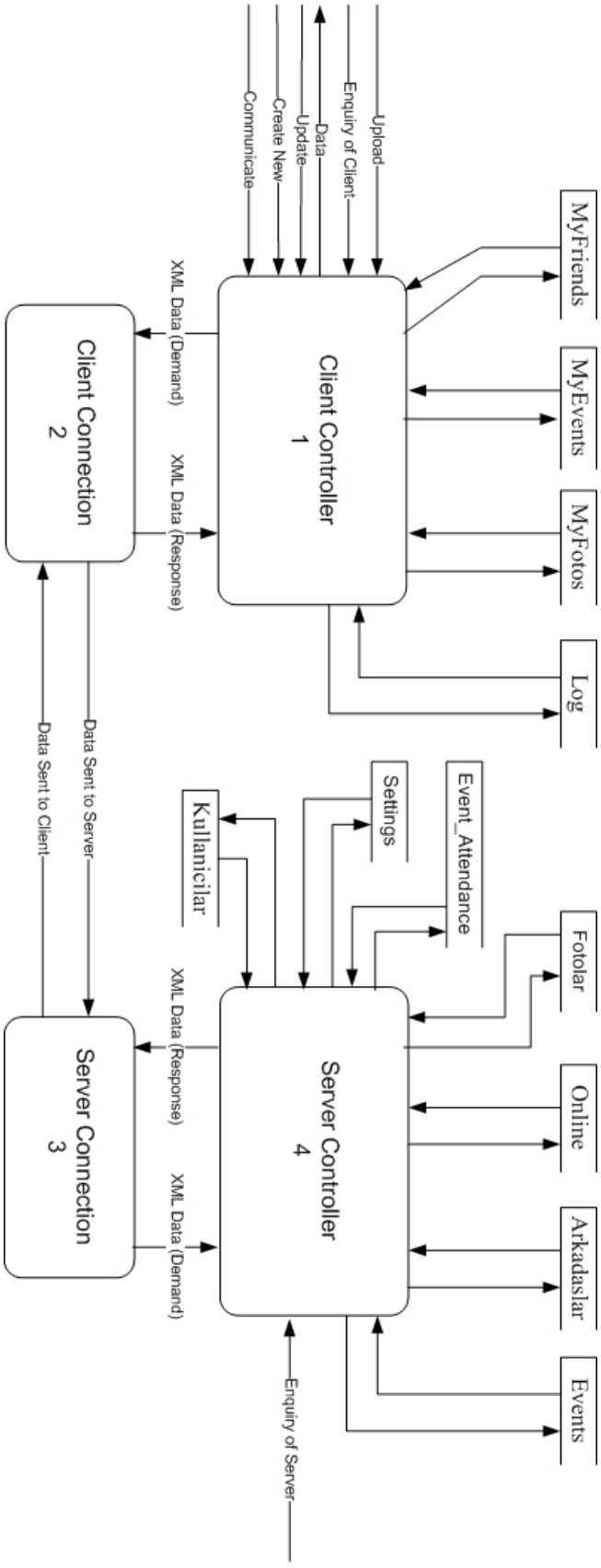
measureDistance: Measures distance between other objects.
isInside: Is our coordinate inside the rectangle coming as a parameter.
move: Change coordinates referenced to saved coordinates.

5.3 Data Flow Diagrams

5.3.1 Context Level Diagram

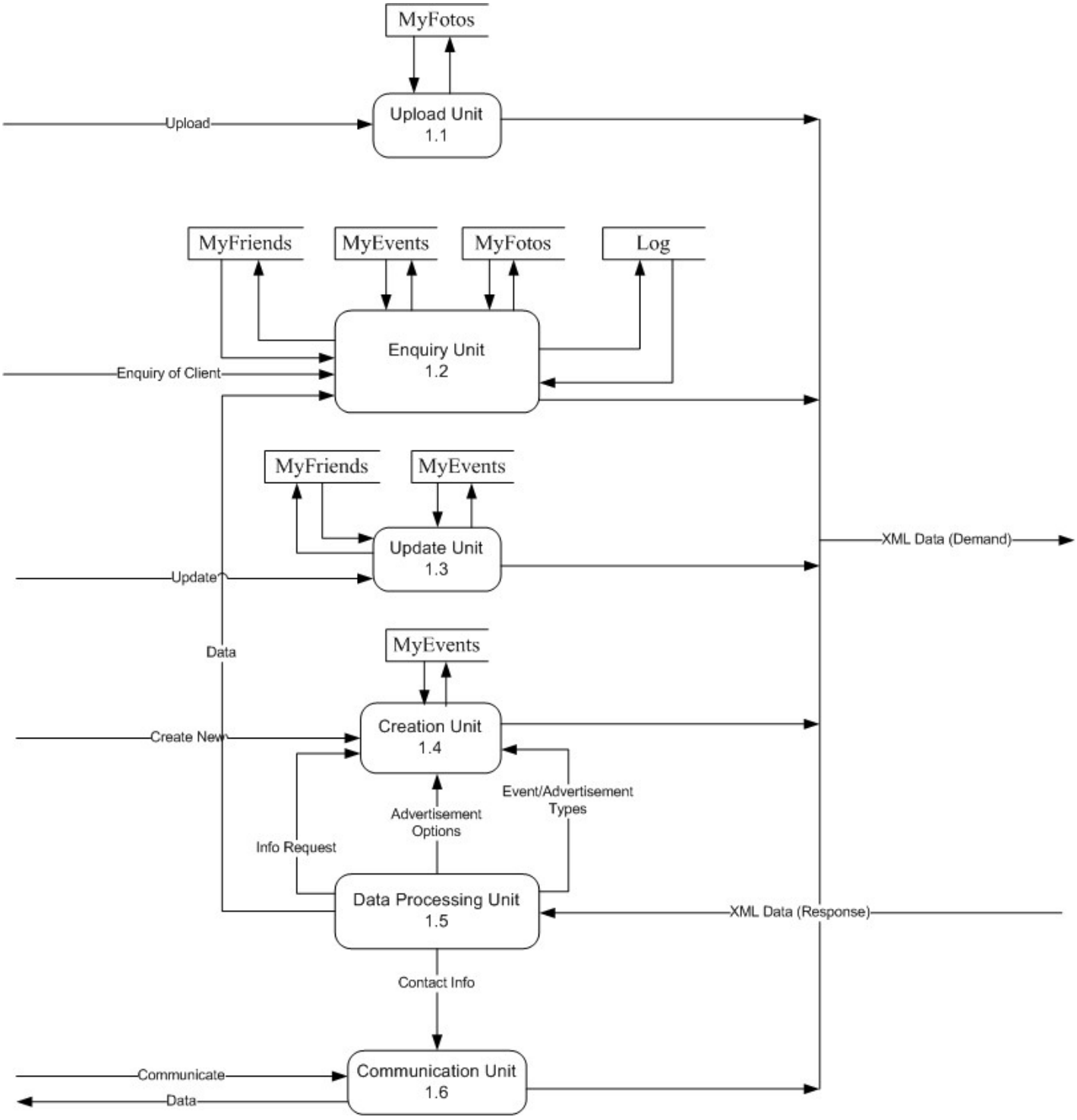


5.3.2 Level0 DFD

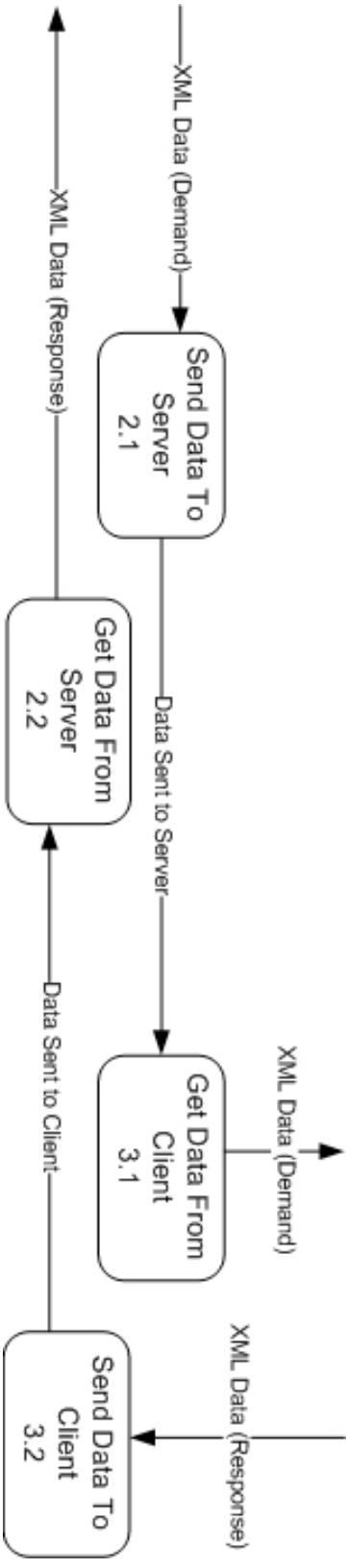


5.3.3 Level1 DFD

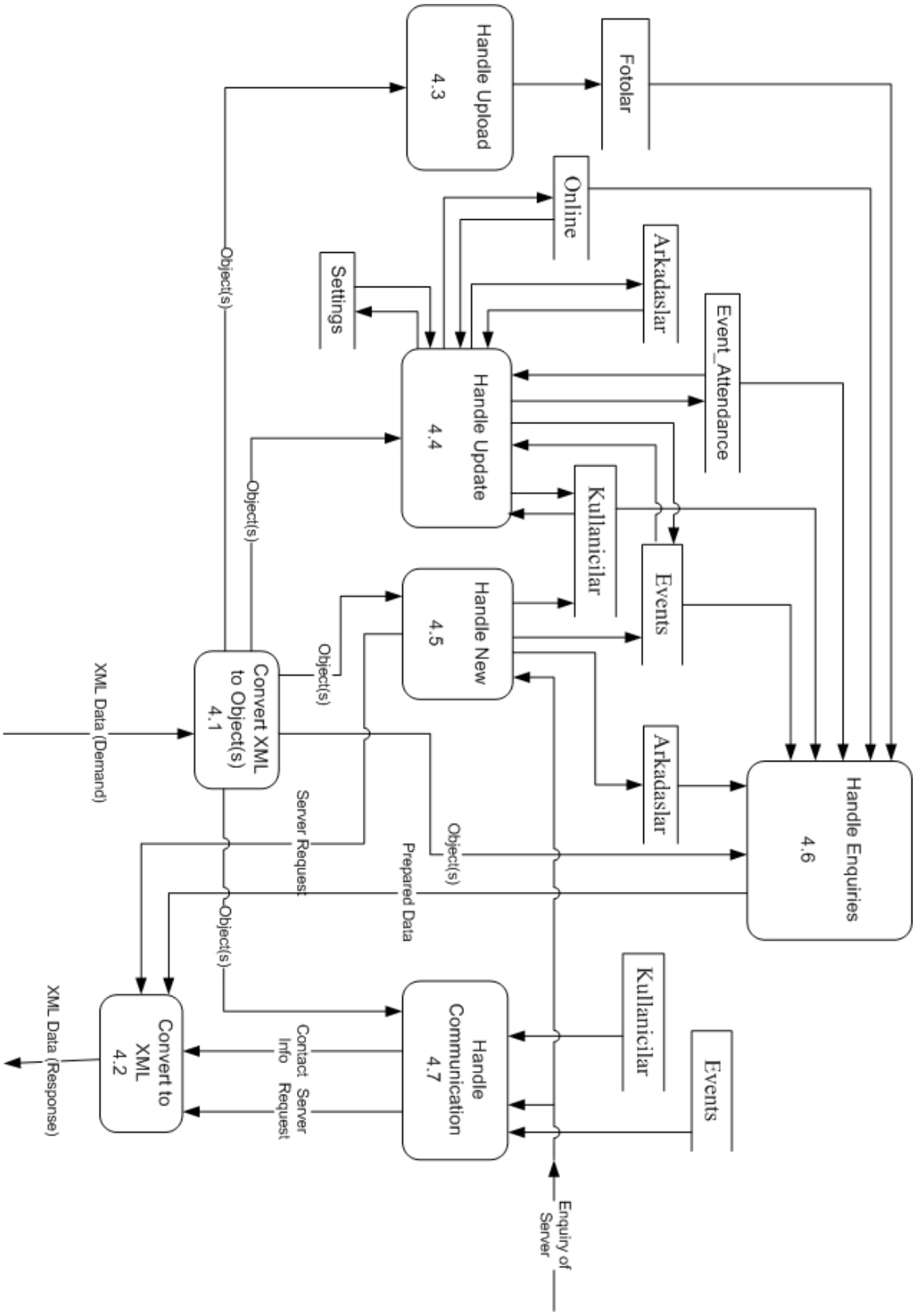
1. Client Controller



2. Client Connection & Server Connection

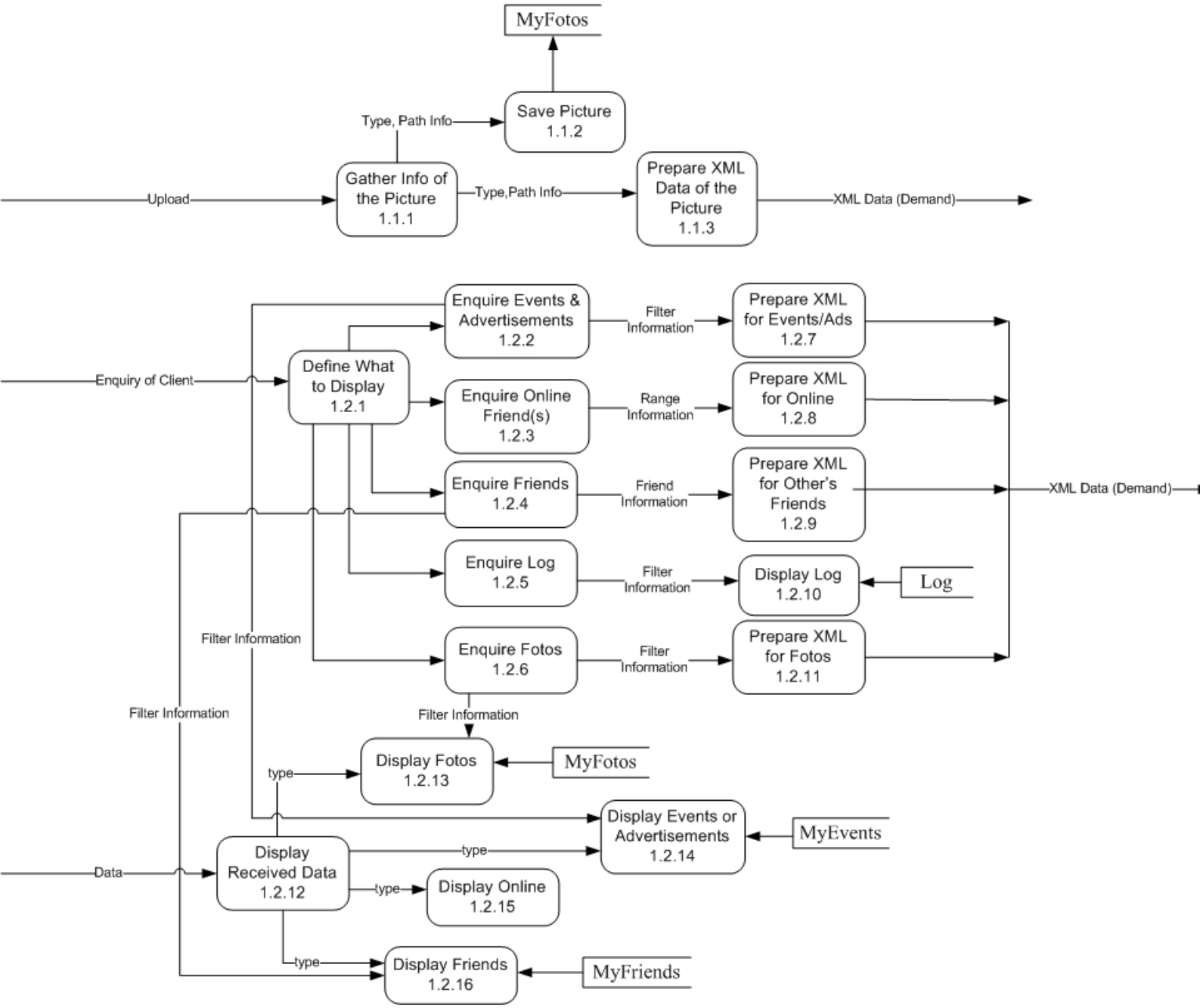


3. Server Controller



5.3.4 Level2 DFD

1. Client's Upload Unit & Enquiry Unit



Name	Type, Path Info
From	Gather Info of the Picture 1.1.1
To	Save Picture 1.1.2 & Prepare XML Data of the Picture 1.1.3
Description	Includes type and path information of the picture

Name	Filter Information
From	Enquire Events & Advertisements 1.2.2
To	Prepare XML for Events/Ads 1.2.7 & Display Events or Advertisements 1.2.14
Description	Includes information of what to filter from Events or MyEvents table

Name	Range Information
From	Enquire Online Friend(s) 1.2.3
To	Prepare XML for Online 1.2.8
Description	Includes range information to be filtered from Online Friends

Name	Friend Information
From	Enquire Friend(s) 1.2.4
To	Prepare XML for Other's Friends 1.2.9
Description	Includes friend information (i.e. name, id) to be filtered from Arkadaslar table

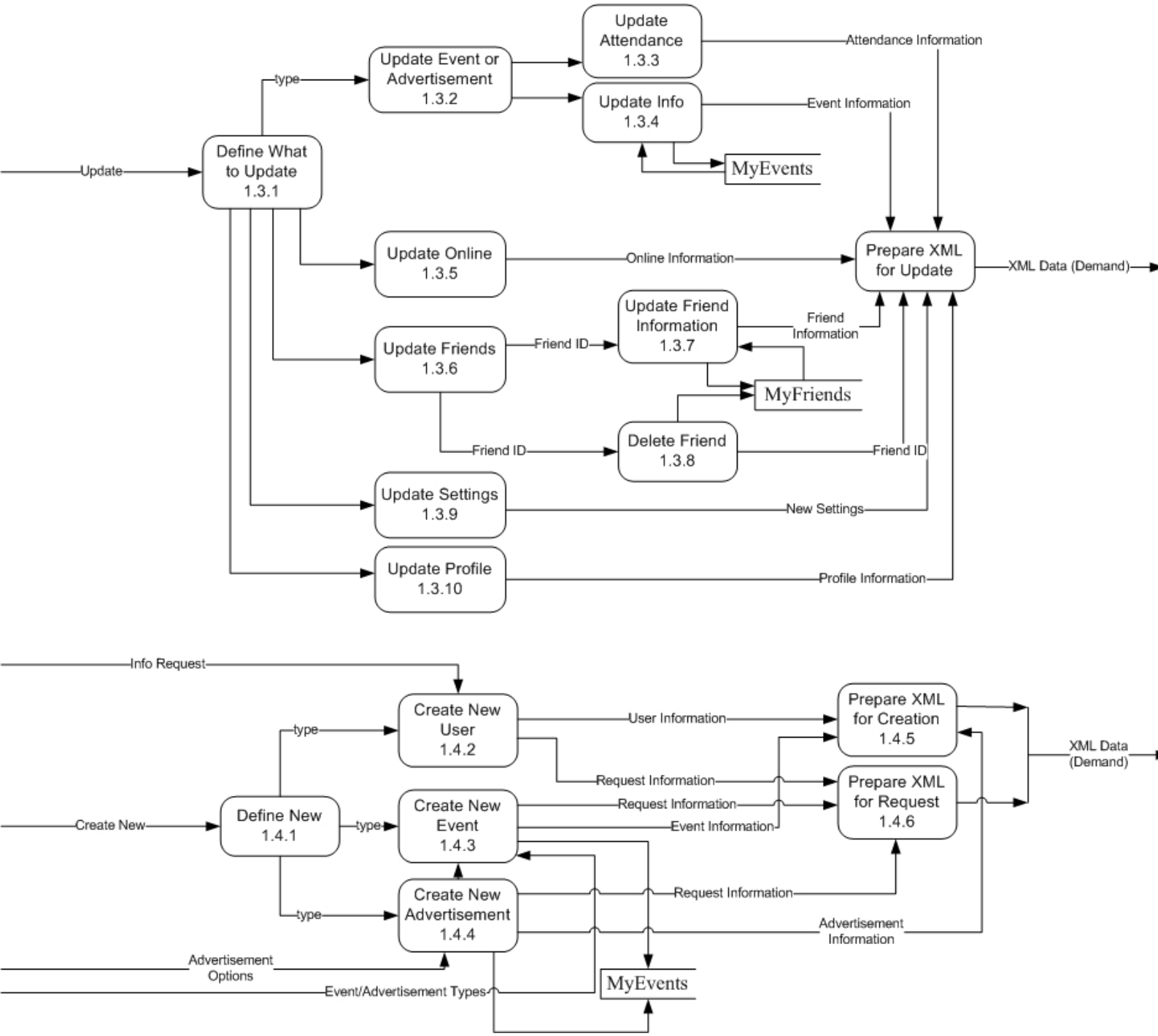
Name	Filter Information
From	Enquire Friend(s) 1.2.4
To	Display Friends 1.2.16
Description	Includes friend information (i.e. name, id) to be filtered from MyFriends table

Name	Filter Information
From	Enquire Log 1.2.5
To	Display Log 1.2.10
Description	Includes log information (i.e. time) to be filtered from Log table

Name	Filter Information
From	Enquire Fotos 1.2.6 & Display Fotos 1.2.13
To	Prepare XML for Fotos 1.2.11
Description	Includes foto information (i.e. type) to be filtered from Fotolar or MyFotos table

Name	type
From	Display Received Data 1.2.12
To	Display Fotos 1.2.13 & Display Events or Advertisement 1.2.14 & Display Online 1.2.15 & Display Friends 1.2.16
Description	Includes type of received data

2. Client's Update Unit & Creation Unit



Name	type
From	Define What to Update 1.3.1
To	Update Event or Advertisement 1.3.2 & Update Online 1.3.5 & Update Friends 1.3.6 & Update Settings 1.3.9 & Update Profile 1.3.10
Description	Includes type of data to be updated

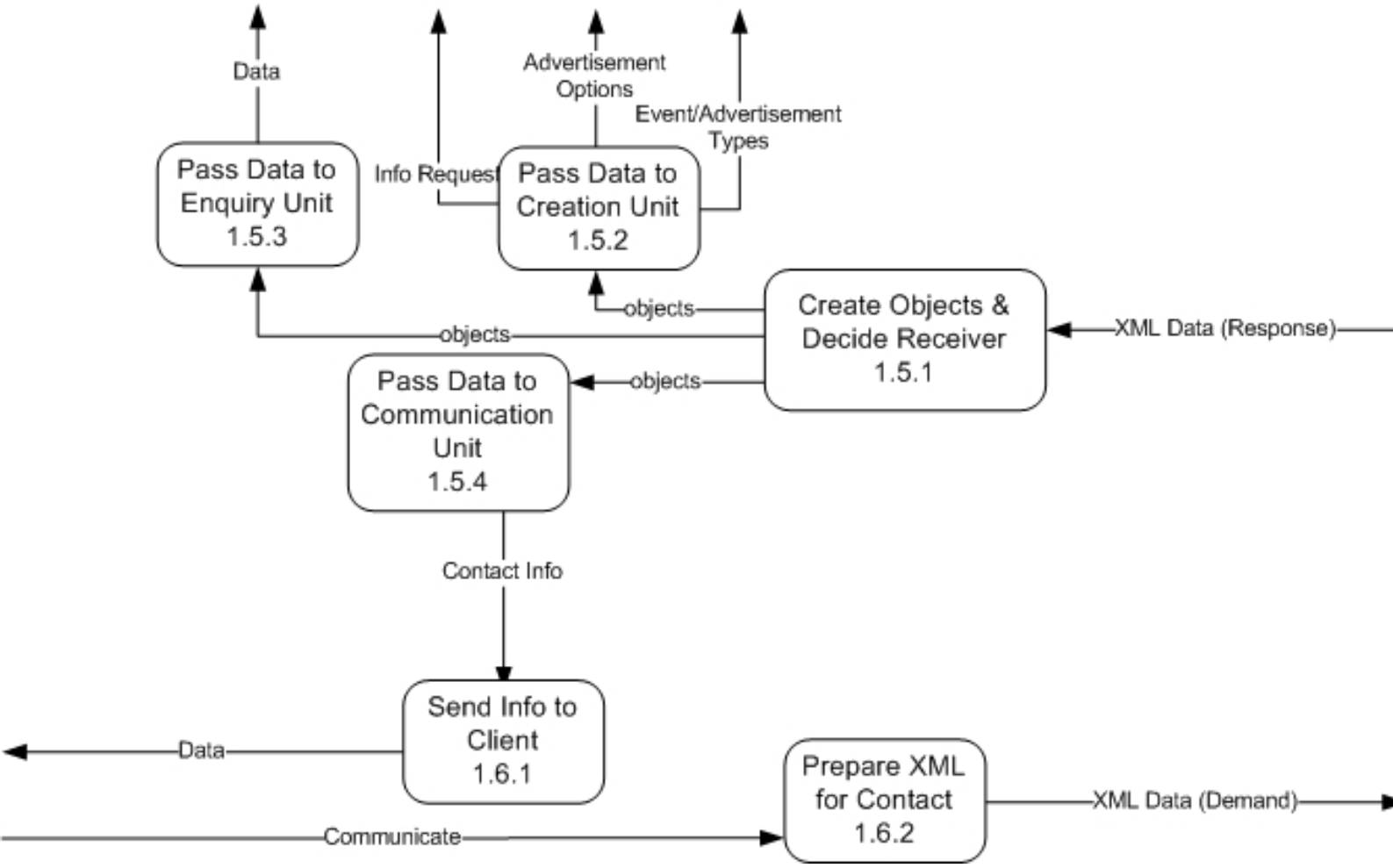
Name	type
From	Define New 1.4.1
To	Create New User 1.4.2 & Create New Event 1.4.3 & Create New Advertisement 1.4.4 &
Description	Includes type of data to be created

Name	User Information
From	Create New User 1.4.2
To	Prepare XML for Creation 1.4.5
Description	Includes full information of the user for creating new user

Name	Request Information
From	Create New User 1.4.2
To	Prepare XML for Request 1.4.6
Description	Includes basic information of the user for creating new user to be confirmed by the server

- These last two tables are similar for create new event/advertisement tools.

3. Client's Data Processing Unit & Communication Unit



Name	objects
From	Create Objects & Decide Receiver 1.5.1
To	Pass Data to Creation Unit 1.5.2 & Pass Data to Enquiry Unit 1.5.3 & Pass Data to Communication Unit 1.5.4
Description	Includes objects created from received XML data

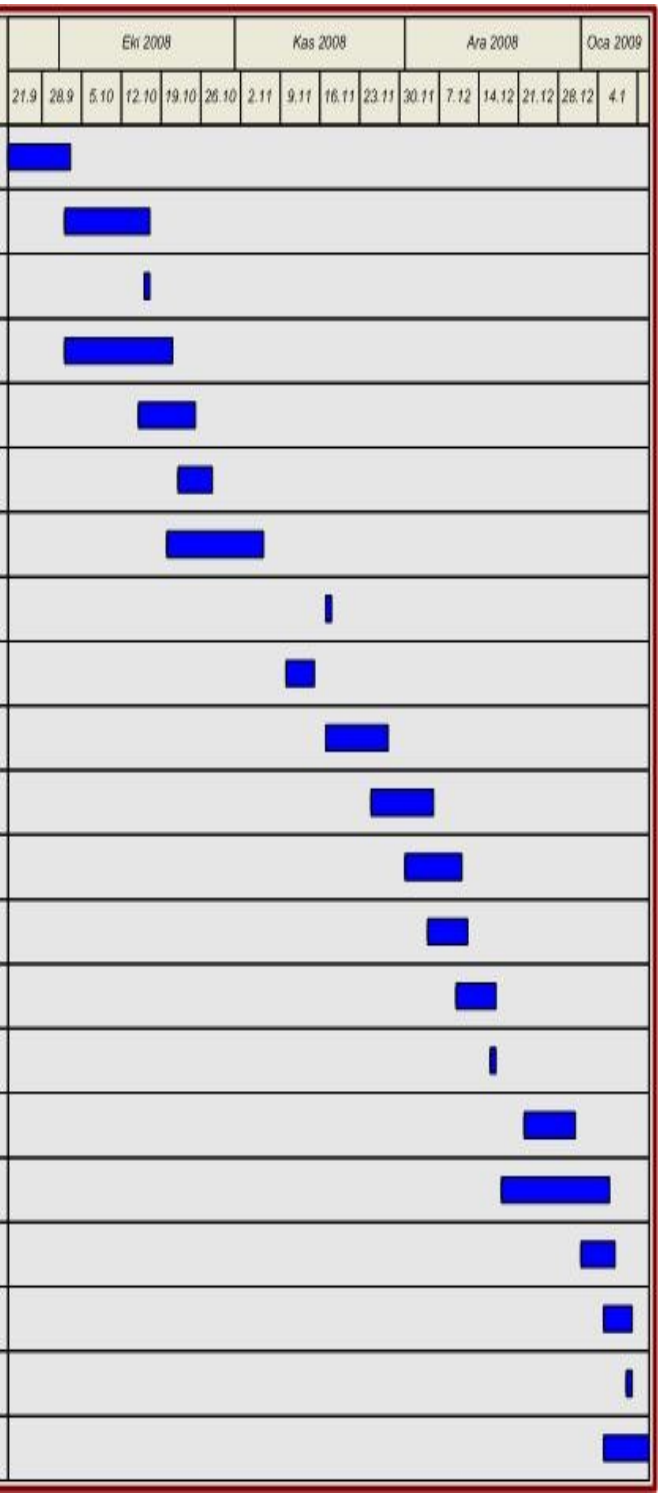
Name	Contact Info
From	Pass Data to Communication Unit 1.5.4
To	Send Info to Client 1.6.1
Description	Includes contact information (i.e. phone no,e-mail) to be send to client

6. DESIGN CONSTRAINTS

The schedule has milestones which are synchronized with reports and the implementations. The current stage of the project can be viewed from the chart. Also the roadmap is discussed in the future work.

6.1 Gantt Chart

ID	Task Name	Start	Finish	Duration
1	Teaming up & Project Topic	22.09.2008	02.10.2008	1,8w
2	Preparing the Proposal Report	02.10.2008	16.10.2008	2,2w
3	Milestone 0	16.10.2008	16.10.2008	,2w
4	Survey About the Communication Market	02.10.2008	20.10.2008	2,6w
5	Survey about new communication technologies	15.10.2008	24.10.2008	1,6w
6	Requirement Analysis	22.10.2008	27.10.2008	,8w
7	Survey about Android	20.10.2008	05.11.2008	2,6w
8	Milestone 1	17.11.2008	17.11.2008	,2w
9	Preparing Requirement Analysis Report	10.11.2008	14.11.2008	1w
10	Creating Development environment	17.11.2008	27.11.2008	1,8w
11	Getting used to Eclipse + Android	25.11.2008	05.12.2008	1,8w
12	Survey on Gtalk and Google Maps	01.12.2008	10.12.2008	1,6w
13	Review and preparing for Report	05.12.2008	11.12.2008	1w
14	Preparing Initial Design Report	10.12.2008	16.12.2008	1w
15	Milestone 2	16.12.2008	16.12.2008	,2w
16	Combining existing technologies	22.12.2008	30.12.2008	1,4w
17	Try to complete some features of the Product	18.12.2008	05.01.2009	2,6w
18	Redesigning and progress analysis	01.01.2009	06.01.2009	,8w
19	Final Design Report	05.01.2009	09.01.2009	1w
20	Milestone 3	09.01.2009	09.01.2009	,2w
21	Preparing Prototype Demo	05.01.2009	12.01.2009	1,2w



6.2 Future Work

Future work exists for both surveying the subjects and implementation.

6.2.1 Survey Topics

Since the project has started, Fizan has come a long way but many things are yet to be done. The team decided to survey the subjects in an evolutionary approach.

First of all from now on, the team has to familiar with java technology, learning how to write enterprise level applications is complex work. Moreover, in the enterprise level development, accessing and using database and connecting to server or client, sending and requesting messages have priority in learning process.

Secondly, using android sdk while developing the program for cell-phone client side is extremely important. Though the application has web and cell-phone clients, dominant one is cell-phone client, and this situation increased importance of android sdk.

Although, some research has been done on embedding Google map and taking location data, using it with android, it is not enough to construct such a project so, more research should be done in this area.

Finally, for the web part no research has been made so far, team has to start to make research on web part. Especially, Java servlets would be helpful.

6.2.2 Implementation

Current state of implementation is in the beginning stages. Sample codes have written for testing available technology and environment that we use in developing is determined and tested so far.

Since the team currently lacks knowledge in android, project's component design has been staggered and in the overall the current component designs are simple and idealistic. They must be improved in the light of feedbacks from the milestones which are already scheduled.

Another important issue in implementation is, dividing the work equally and synchronization between team members. In order to get rid of these kinds of problems we should use versioning tools while developing the code.