BILKENT UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

CS411 Software Architecture Design Final Project

Group 5
Active Citizen

Hilal Güler
İhsan Mert Özçelik
Beyza Nur Kökcan
Mustafa Ozan Karsavuran
27/12/2011

Table of Contents

1. Introduction

As citizens, we face difficulties and problems occurring in our cities, which is needed to take actions by the local municipalities. When citizens witness difficulties, they want to direct their messages to the officials to solve the problem. However, citizens generally ignore the problem because they cannot communicate the officials easily. When a problem related to cities and municipalities occurs, citizen should play active role to address the problem. Otherwise, if the management of the problem is processed by only the related municipality, the solution to the problem can be delayed or ignored. The best way to solve this problem is to provide an application which will work on smart phones that most of the people have nowadays and allow the citizen to direct their complaints to the related municipalities without bothering with procedures and immediately when they face the problem.

Active Citizen Application will work on smart phones and help the citizens actively participate in the city management and the city life. In this way, citizens, non-governmental organizations and municipalities will be able to communicate quickly to solve the problem. Citizens will be able to make complaints or suggestions on topics related to municipal services and will be able to actively join the discussions via the help of our system. They will also have a chance to express their admirations about the nicely-managed parts of the city. Our proposed system will provide online solutions to the problem of making citizens' voice to be heard by the municipality administrations. We propose a system that would considerably decrease the time and money to be spent for reaching the necessary officials. By the help of the application, the ongoing problems in the cities will be able to be detected and solved in a quicker way. The application will also provide ease of access to administrative institutions. Instead of going to the administrative buildings and filling in an application form and waiting for their applications to be read and verified, citizens can make their comments or requests and check the application status online. In terms of these, citizens will be more active in participating the city management and city life. The availability of the application in a mobile device would make it straightforward for the users to communicate with the associated offices about a problem and thus, increasing the chance of reaching a solution.

The system provides different user interface to different user types. Citizens will be able to register and later log on to the system. Additionally, people will be able to enter the system as guest users with limited functionality. By choosing from different categories available such as environment, health services, transportation services, construction services, social services, the user will be directed to different menus and pages that include announcement section, suggestions/complaints section and the discussion section. Guest users will be able to read announcements, complaints, and discussions; however, they will not be able to give any input. Registered citizens who log in will be able to make comments on the problems they faced in their location. These inputs will be evaluated by local municipal officials. Citizens will follow the situation of their complaints or suggestions like pending, being evaluated, or accepted/rejected. In addition, they will be able to join discussions and make comments about an on-going situation in their cities. Via the application, the citizens can also see the group list and join the ones he or she wants to. In general, the system allows the citizens to be able to

participate in the city management in active manner. The system will be developed to be used by the cities in Turkey; therefore the system will be developed in Turkish.

In the rest of the report, initially the case description will be made in section 2 and the software architecture design process applied throughout the project will be described in general in section 3. After these parts, in section 4 requirement analysis which includes the stakeholder description, textual requirements, context diagram, use case models, textual use case description, architectural scenarios, user interface examples and dynamic models are described. Then, in section 5 technical problem analyses will be described in detail. Finally, domain analysis including domain solution, knowledge sources and domain concepts and software architecture design will be explained in section 6 and section 7. At the end of the report, you can find the brief conclusion which summarizes the whole process and mentions about the future work in section 8.

2. Case Description

Throughout the day, we meet lots of deficient sides of the municipal management. Most of the time, as citizens, we complain about these deficient sides which make our lives harder, but we do not take any action. The reasons behind this are the working hours of the municipalities and also some tedious procedures that should be applied during the submission of the complaints. On the other hand, the complaints may not be answered in a convenient manner as well. The citizen should be determined on his/her action. However, people work during the day and once they find spare time, they do not want to bother with those issues. The citizens make complaints verbally, i.e. no official action is taken and the problems with the city remain as they are. Additionally, the current systems where the citizens offer problems in their municipalities do not meet the active participation of citizens in city management because they can put time and place obstacles in citizens' way. When they face a problem, citizens would want to deliver it to the officials just by the time they meet the problem. However, using current systems, they can deliver their problems after they face the problem, which can lead citizens to ignore delivery of their messages to the officials and take necessary actions.

For these reasons, the main goal of the project is to provide active participation in city management for citizens so that the obstacles or difficulties can be solved with less effort and within short time. Therefore barriers between citizens and officials can be removed and citizens, non-governmental organizations and municipalities will be able to communicate quickly and easily. With this goal, capability and quality of the current system will be improved and reduce the total cost necessary for solving problems.

Additionally, the project has a goal to improve municipality officials' motivation so that the city order can be improved in a better way. When citizens report their complaints, officials get to work to turn the situation into their advantages. If the problem that citizen reports is solved, the rating of the municipality could be get higher. Also when the citizens offer their opinions

and suggestion, municipality can directly address the citizens' offer so this could also affect their ratings.

As a result, the project has goals to;

- Improve active participation of citizens in the city management
- Reduce the total cost to providing better management of the city according to citizens' needs.
- Have better improvement than existing systems in terms of capability, performance, product lines, ease of use, security, safety and functionality.
- Have better improvement than the current systems in terms of cost including development cost, maintenance cost and deployment cost.
- Improve maintain the city order by motivating municipality officials and make workflow in the municipalities stronger.

3. The software architecture design process

The software architecture design process of the project is mostly organized according to the synthesis-based software architecture design approach, which includes three phases: technical problem analysis, domain analysis and alternative space analysis. Among these three phases, alternative space analysis will not be applied to the system. Before technical problem analysis, context diagram of the system as the first ingredient of architectural information presented to a reader as a general view of the system and in order to show how the system under consideration interacts with the outside world together with the interactions between the existing sub-systems. Context diagram is useful for understanding the external environment within which the system must operate and provides general understanding of the whole system.

Technical problem analysis in the process aims at specifying, identifying, determining, prioritizing sub-problems to map these sub-problems to solution domains. In identifying the problem, abstraction is made from concrete requirements to comprehend the essence of the problem; application domain and quality requirements such as robustness, reusability, various trades-off and time constraints are analyzed. In this way, systematic analysis of the problem is done and solving wrong problem is avoided.

In domain analysis phase, there will be two main steps, which are domain scoping and domain modeling. In domain scoping, for each sub-problem of technical problem analysis, solution domains are identified and prioritized; then for each domain, knowledge sources are identified and prioritized. In domain modeling, domain concepts are extracted from each knowledge source; then domain concepts are structured and refined.

4. Requirements Analysis

4.1.Stakeholders

The stakeholders can be divided into two groups. First group is the direct stakeholders and include:

- Citizen
- Local municipality
- Non-governmental organizations
- System Administrator

The first group of the stakeholders is end users of the system. They are the actual users of Active Citizen but their roles in the system differ among each other.

The second group of stakeholders of the system is the ones who take care of the technical development of the system which participate in the production of the system but they are not the end users. Second group include:

- Database Administrator
- Software Architect
- Software Designer
- System Maintainer
- Implementer
- System Integrator
- Project Manager
- System Engineer
- Tester

Database Administrator: This stakeholder is responsible for database storing including optimization, design, security etc.

Software Architect: This stakeholder is responsible for documenting architecture of the system.

Software Designer: Responsible for software design of the architecture meeting requirements.

System Maintainer: Responsible for enhancement to the system.

Implementer: Responsible for implementation of specific elements according to requirements, design and architecture.

System integrator: Responsible for the integration of parts to the system.

Project Manager: Responsible for managing the project including planning, scheduling, delivering components to integration.

Software Engineer: Responsible for design and development of system or component of the system.

Tester: Responsible for test and verification of the system and parts of the system.

4.2. Textual requirements

Textual requirements are derived according to the stakeholders' needs and they are listed below.

1. Citizens shall be able to create new accounts and log on to the system with their accounts.
2. They shall be able to make comments on the problems they faced in their location.
3. They shall express their opinion about anything related to municipalities.
4. Users shall be able to upload picture to their comments and suggestions.
5. Users shall also pose problems about the municipalities and shortages that they face throughout the day and upload photographs as evidence to their complaints.
6. They shall also follow the situation of their comments like pending, being evaluated, or replied.
7. Citizens also shall be able to look through and comment on the other comments made by other users.
8. Citizens shall be able to offer suggestions; other users can present their opinions about the suggestion. All these suggestions will be evaluated by officials.
9. Users shall rate each unit of the municipal government besides their comments using "like" and "dislike" buttons.
10. Local municipality officials shall be able to log on to the system with their accounts determined by system administrators.
11. Local municipality officials shall access all comments and suggestions made by citizens.
12. They shall be able to decide whether a comment related and worth to be evaluated or not.
13. In any case of abuse, officials shall be able to report the users who abuse their rights to system administrators.
14. Administrators shall evaluate the notifications made by officials about citizens.
15. They shall be able to warn them and delete their account if it is needed.
16. Officials also shall be able to reply the comments and suggestions after evaluation.
17. System administrators shall be responsible for creating new accounts for local municipal officials. They control all accounts.
18. Citizens shall be able to make complaint about officials, in such cases administrators shall evaluate these complaints and they can take the necessary actions.
19. Non-governmental organizations shall be able to log in to the system as the other users, form groups and announce their social actions via the system.

20. “Active Citizen” system shall offer its users to make comments about some other public areas like restaurants, shopping centers and cinemas. These comments can be for informing other users about the quality of those places.
21. Moreover, users shall be able to present new places.
22. Citizens and non-governmental organizations can attend surveys delivered by the system.
23. Officials can view the complaint/suggestion statistics related to one municipality or the city.
24. The system shall provide FAQ (Frequently Asked Questions) pages, assigning performance ratings to each unit of municipal government according to feedback received from the users directly.

4.3. Use case model

The use case models of the system for each user of the system including citizen, non-governmental organizations, local municipality officials and the system administrator are included.

In Figure 1, use case diagram for guest is shown. Guest can login, access opinions and view the current 10 actions.

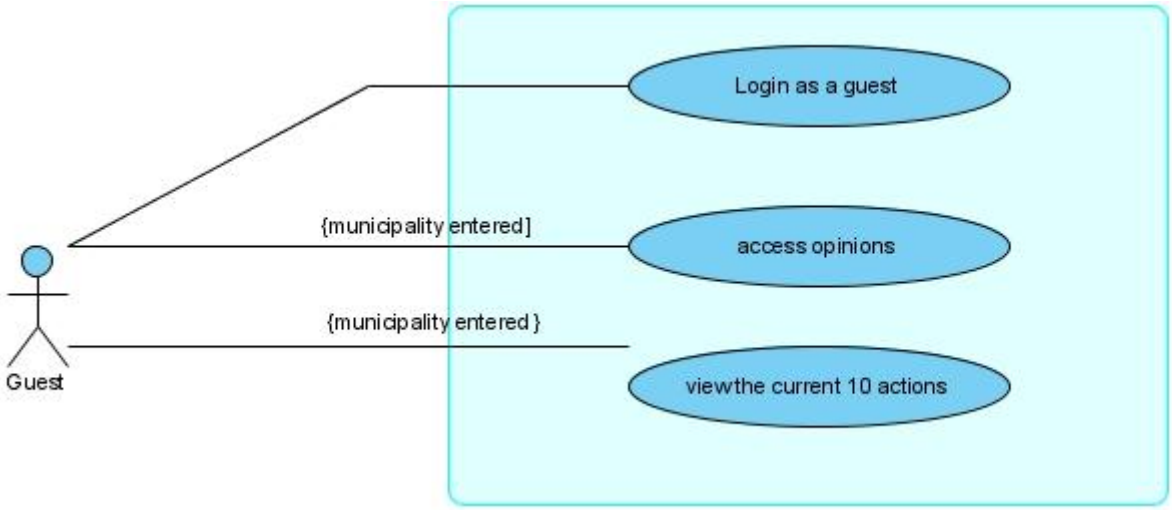

Figure 1: Guest Use Case

Use case diagram for citizen could be seen in Figure 2. Operations are basically combined in Participate administration case with include relationship. As shown in diagram, citizen can express opinions including reporting complaint, offering suggestions, making comments and starting discussions as uploading pictures. Citizen can also follow of his/her opinions, rate other users’ comments and attend surveys.

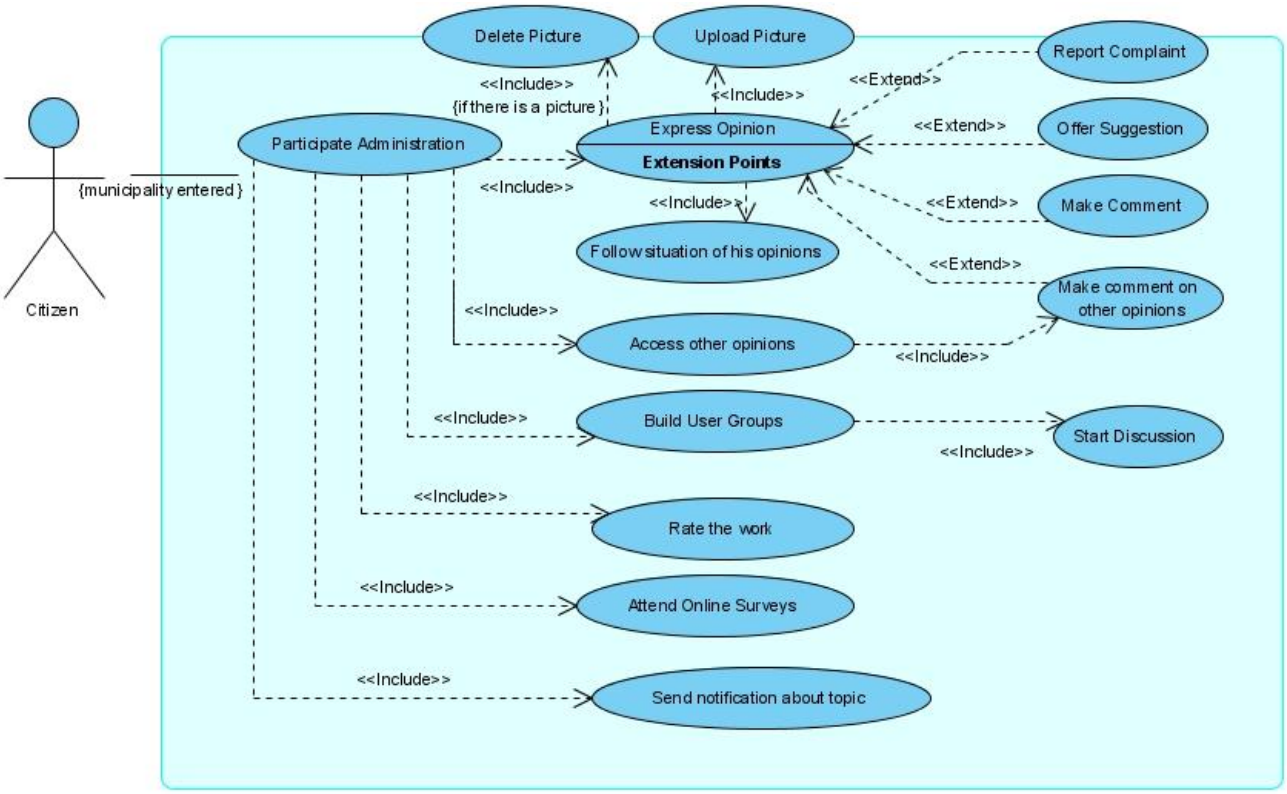

Figure 2: Citizen Use Case

Use case diagram for representatives of non-governmental organizations is shown in Figure 3. Since citizen can also manage account it is repeated as actor.

Organization representative can announce social actions to the citizens and manage user groups. While managing accounts, citizen can create an account and login including entering username, password, verifying id.

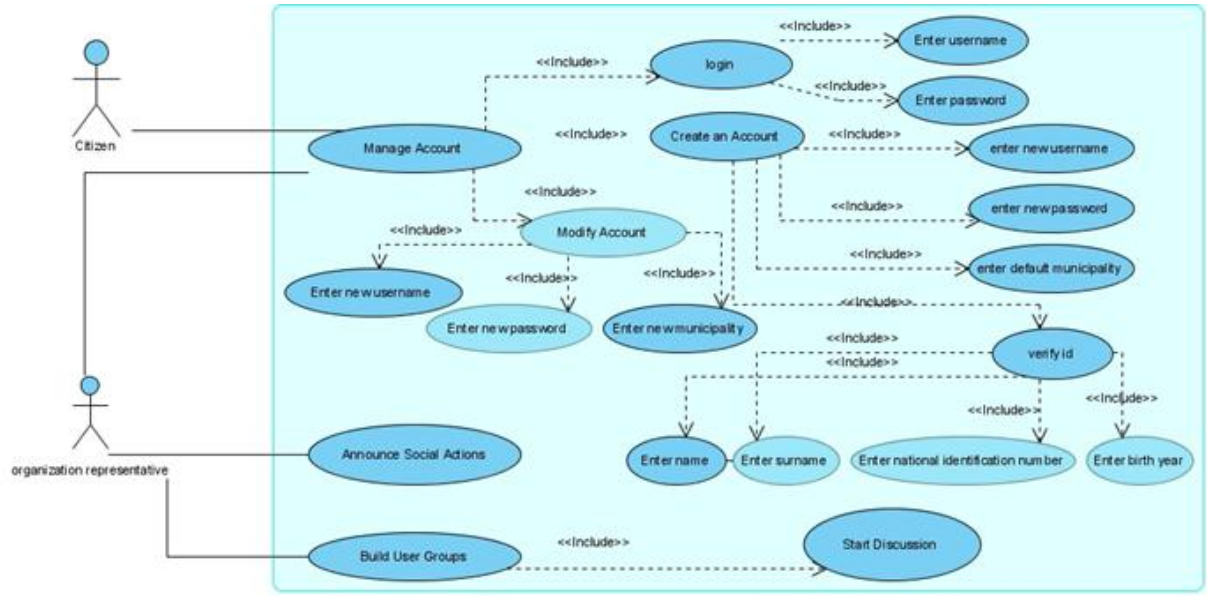

Figure 3: Non-governmental organization use case

In Figure 4, use case diagram for local municipality officials is described. Official can login the system by entering username and password as well as access opinions by changing the status, replying and rejecting the comments.

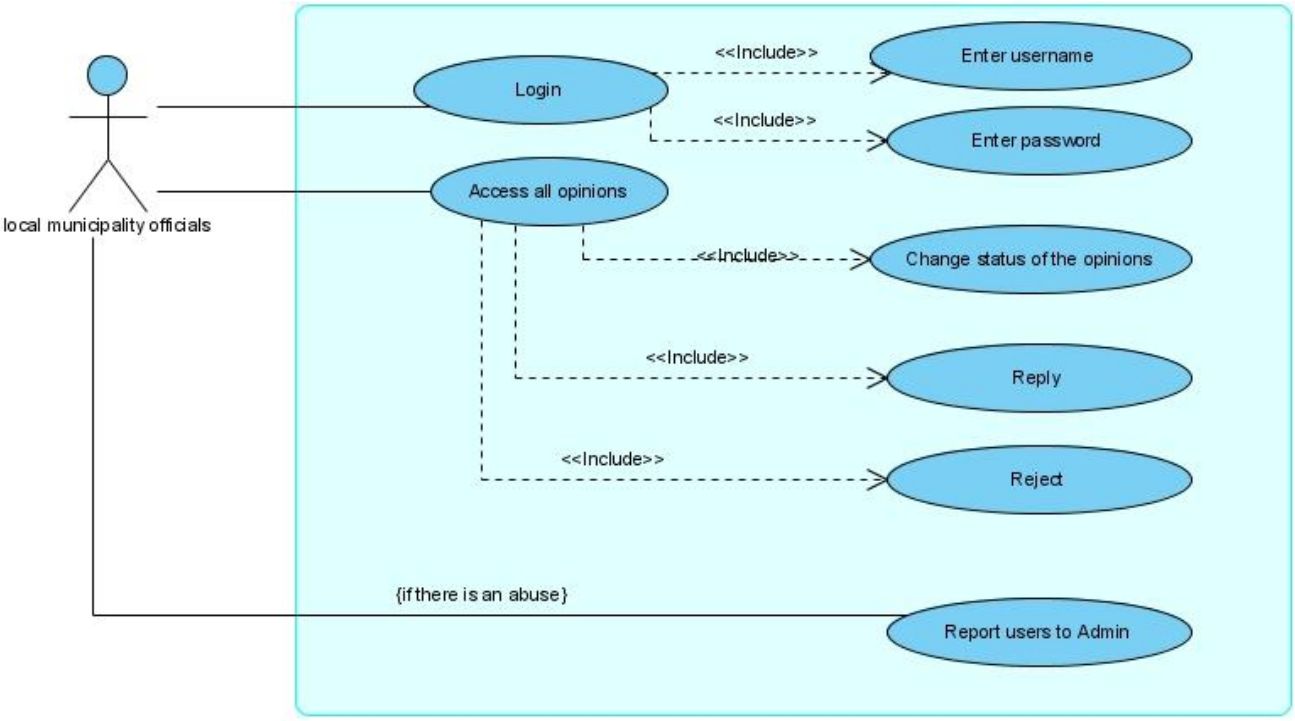

Figure 4: Local Municipality Use Case

Operations that are available to admin are shown in Figure 5. Admin can manage their accounts and add local officials to the system. Admin can also evaluate reports from officials and thereby send warning to the citizen or ban.

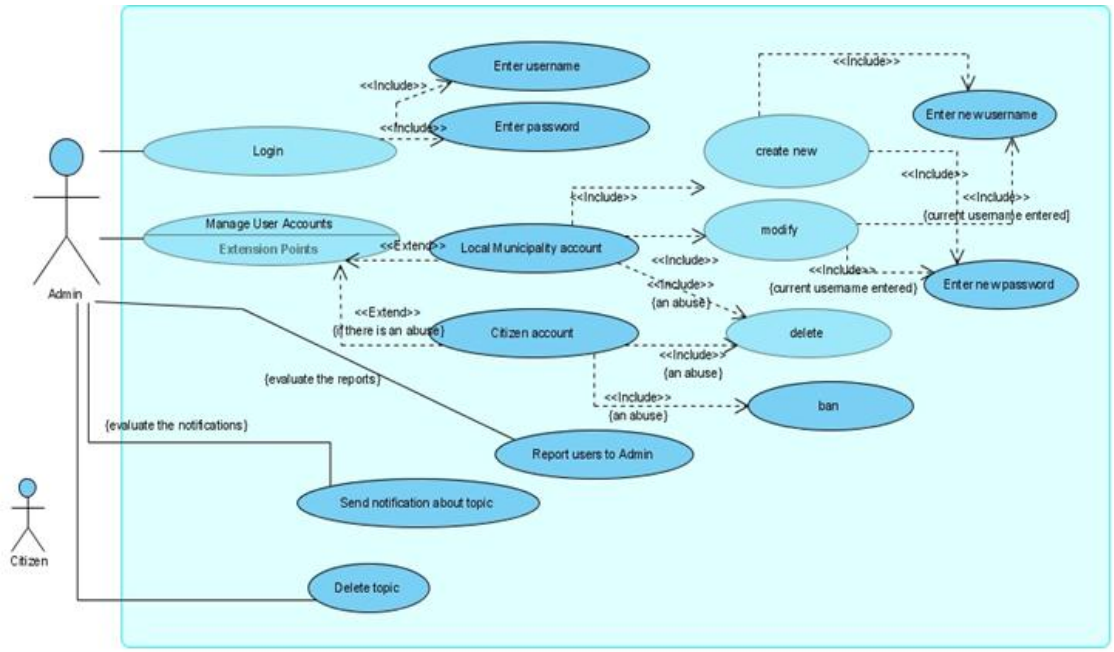

Figure 5: Admin Use Case

4.4. Textual Use Case Descriptions

Use case name	Entering as a Guest User to the system
Primary Actor	Citizen
Precondition	The user is an anonymous user.
Flow of events	1. Citizen starts the application. 2. In the login screen, citizen chooses “Misafir Girişi” option. 3. Citizen enters to the system as a guest. 4. Citizen chooses the city and the municipality s/he wants to get information about from the initial screen. 5. Citizen chooses among the categories either by using quick menu option or by choosing sub categories one by one. 6. Citizen views the current 10 actions of the municipality s/he has chosen. 7. Citizen chooses one of the actions. 8. Citizen views the comments that are written about this action. 9. Citizen logs out the system.
Extensions	*a. At any time, user can log out from the system. 6a. At any time, user can quit the system.

Use case name	Sharing Options and Rating the work as a citizen
Primary Actor	Citizen
Precondition	The user is a logged in user. The user is on the main page.
Flow of events	1. Citizen can change the city and the municipality that s/he wants to get information about from the initial screen. 2. Citizen chooses among the categories either by using quick menu option or by choosing sub categories one by one. 3. Citizen views the most currently ongoing works and chooses one of them. 4. Citizen writes her/his opinion about the work and gives a suggestion about how the work should be conducted in order not to aggrieve the citizens. The citizen may mention about a possible problem in order to stress the necessity of the precaution that should be taken. 5. Citizen submits her/his comment. 6. The citizen can also rate the work using “like” and “dislike” icons even without making any comment. The likes and dislikes are used by the system in order to rate the local municipalities. 7. Citizen logs out the system.

Extensions	*a. At any time, user can log out from the system.

Use case name	Reporting complaints and adding photographs as evidence
Primary Actor	Citizen
Precondition	The user is a logged in user. The user is in the "Şikayetler" tab.
Flow of events	1. The citizen can change the city and the municipality that s/he faced with the problem from the initial screen. 2. S/he can see that there is an open pit and there is no precaution is taken. 3. S/he takes a photograph of the open pit with her/his smart phone. 4. S/he adds the photograph and writes a complaint about the situation. 5. S/he submits the complaints and sees that its situation is "pending". 6. S/he submitted the complaint successfully and the complaint is sent to the local municipal officials in order to be evaluated. 7. S/he logs out the system.
Extensions	*a. At any time, user can log out from the system.

Use case name	Banning and deleting user account – After warning once
Primary Actor	Admin
Precondition	1. Admin user wants to ban and delete a citizen user account. 2. Admin user is logged in to the Active Citizen Application and in the "Kullanıcı Raporları" tab of it.
Flow of events	1. Admin user sees a list of the notifications sent by the local municipal officials and citizens. 2. Admin user selects which notification he or she would like to read. 3. Admin user reads the details of the specific notification. 4. If the admin finds out that the citizen or local municipal official is not obeying the rules of the system, he or she sends warning initially. 5. If the problem arrives again, then the admin bans the user account from entering the system or deletes his or her account completely. 6. System asks for confirmation and gives feedback about whether the account is banned / deleted, or a problem occurred during the action.
Extensions	*a. At any time, user can log out from the system.

Use case name	Replying a complaint or a suggestion
Primary Actor	Local municipal official
Precondition	1. Local municipal official user wants to reply a complaint or suggestion made by a citizen. 2. Local municipal official user is logged in to the Active Citizen Application and in the "Öneriler/Şikayetler" tab.
Flow of events	1. Local municipal official user sees a list of complaints or suggestions in descending order of their dates. 2. User chooses one of the complaints or suggestions. 3. User reads and decides to write a reply to the complaint or suggestion. 4. User writes a reply to the complaint or suggestion. 5. After writing his or her respond, the user sends the reply.
Extensions	*a. At any time, user can log out from the system.

4.5. Architectural Scenarios

Architectural scenarios are used in order to increase the understandability and testability of the software architecture. There are 5 architectural scenarios generated for the system below.

Scenario Name: Reporting a complaint

Participating actor: Citizen

Beyza is a user of "Active Citizen". While she is driving, she is stuck on a heavy traffic. She realizes that traffic is heavy because of a big branch of a tree drop on the road. She gets her smart phone and logs in the "active citizen" entering her username and password. She is directed to category page. She selects the "Trafik Hizmetleri" clicking on the road icon. Because she presses the icon long, the sub menu appears. So she can select the "Trafik" and is led to the "Trafik" page directly.

She enters the "Şikayetler ve Öneriler" to ease the traffic. New page appears, she chooses the municipality of where the traffic is slow from the same page clicking on "belediye" and she clicks on the pen icon to open up a new complaint in the list. She takes a photo of the branch with the traffic selecting camera icon and uploads it clicking on "Yükle" then she writes her report. After that, she clicks on the "Gönder" to send her complaint. She stays logged in to follow the situation of her report, which is pending currently. She now waits for her report to be delivered to the local municipality .Her report becomes "Değerlendiriliyor" after a while. Then she realizes that her report is in situation of "Kabul edildi".

Scenario Name: Labeling a complaint as rejected

Participating actor: Local municipality official

İrem logs in the system to read the complaints about a new constructed bridge entering username and password. New page shows up and she clicks on the "Şikayetler ve Öneriler" because she realizes that "Şikayetler ve Öneriler" has "(1)" next to it. System opens up a new page listing the complaints in descending order of their dates showing as unread complaints in a different color. İrem clicks on the unread complaints by "detayları oku" about that bridge to see/expand the rest of the suggestion. She reads the suggestion but decides that it cannot be improved in the way that the citizen requests. She clicks on the "Reddedildi" and new page appears that she needs to commit a reply because she rejects the complaint. She writes why she rejects and clicks on the "Gönder". System asks for confirmation and she says yes, reply is sent. İrem logs out of the system clicking "log out".

Scenario Name: Sharing opinions about the announcement

Participating actor: Citizen

Fatih logs in the application to check the newly opened places entering his username and password. New page where the categories are listed appears. Fatih clicks on "Sosyal Hizmetler" icon. Then the announcements are appeared but he realizes that lists of announcements are for Çankaya because Fatih is registered to Çankaya municipality. He selects the "tümü" from the "Belediye" combo box to see the announcements for all municipalities. And announcements are listed for all. When he is looking up the list of announcements, he is attracted by a restaurant and to read the opinions about it, he clicks on the "detayları oku". He is directed to discussion page. After reading good opinions from other users, he decides to go to that restaurant. He comes to restaurant and orders something. Because the restaurant is new, he cannot trust on in which environment the meats are prepared. He asks to see the kitchen. However, he is disappointed and dissatisfied by how clean the kitchen is. He takes his smart phone and clicks on "Sosyal Hizmetler" icon then he finds the announcement about the restaurant. He clicks on the "detayları oku" and then he selects the pen icon to make a comment on the discussion page. He writes his comment and clicks "Gönder". He logs out of the system clicking "log out".

Scenario Name: Announcing social actions

Participating actor: Non-governmental organization

Cevdet logs in the system as a representative of non-governmental organization entering his username and password. He selects the "Grup Listesi" to announce a new group. He clicks on "Yeni bir grup" to build a user group. He writes the details on the opened page and clicks on the "Gönder". He logs out of the system clicking on "log out".

Scenario Name: Reading Notifications from officials- Ban a user

Participating actor: Admin

Hilal logs in the system as admin entering her username and password. The new page opens up and she clicks on the "Kullanıcı Raporları" button to read the notifications coming from a local municipality official. Notification says that a user writes baseless reports. Hilal clicks on the "değerlendir" to evaluate the report. A new page is opened. She reads the comment that has notified and decides to give a warning to the user. First Hilal labels the notification as "kabul edildi" so that official can follow that user gets warning. Then she deletes the comment clicking on "sil". Hilal selects "Hesapları Yönet" to give a warning the user. She searches the user. Hilal realizes that the user has got one warn before but yet continues to write baseless reports. Therefore Hilal clicks on the "Ban" and bans the user from writing any comments or report. Hilal logs out of the system clicking "log out".

4.6.Prototypes

The label named "Screen1" in all figures between 6 and 13 is a standard label that can be customized for every municipality.

In Figure 6, login screen which is initial screen is shown. User can login as a guest or with account previously created. Another action is register provided on that screen.

Figure 6: Login screen

Figure 7 shows the registration screen of the system. User should provide necessary information and complete registration.

Figure 7: Entering information to the registration screen

Categories provided from municipality are shown as in Figure 8 User can also select another state and municipality or exit from the system by clicking on the button placed at the bottom right corner of the screen.

Figure 8: Categories screen

For example, when user has long-pressed “Çevre” icon in the categories screen a sub-menu will came as shown in the Figure 9.

Figure 9: Environment sub-menu

Continued from Figure 9, if user chose "Çevre Kirliliği" from the previous menu screen, another sub menu will occur as shown in Figure 10.

Figure 10: Environment main screen

As the next step, in Figure 11, user selected "Hava Kirliliği" and wrote complaint along with the photograph.

Figure 11: User has written the complaint and uploaded a photo.

Also, user can read complaints as shown in Figure 12. User can see the state of the complaint like in the screen shot.

Figure 12: Complaint reading screen

Figure 13 shows standard Android sub-menu which will also included in our system. This menu involves shortcuts for the homepage, About section, Frequently Asked Questions section and the logout option.

Figure 13: Android submenu

4.7.State, sequence diagrams

The sequence diagram of the system describes the way and order in which processes operate with one another. The interactions between the objects are shown in a time sequence and the message passings between the classes using the corresponding objects are shown explicitly in the diagram. In Figure 14, the scenario named “Reporting a complaint” from section 4.5 is illustrated.

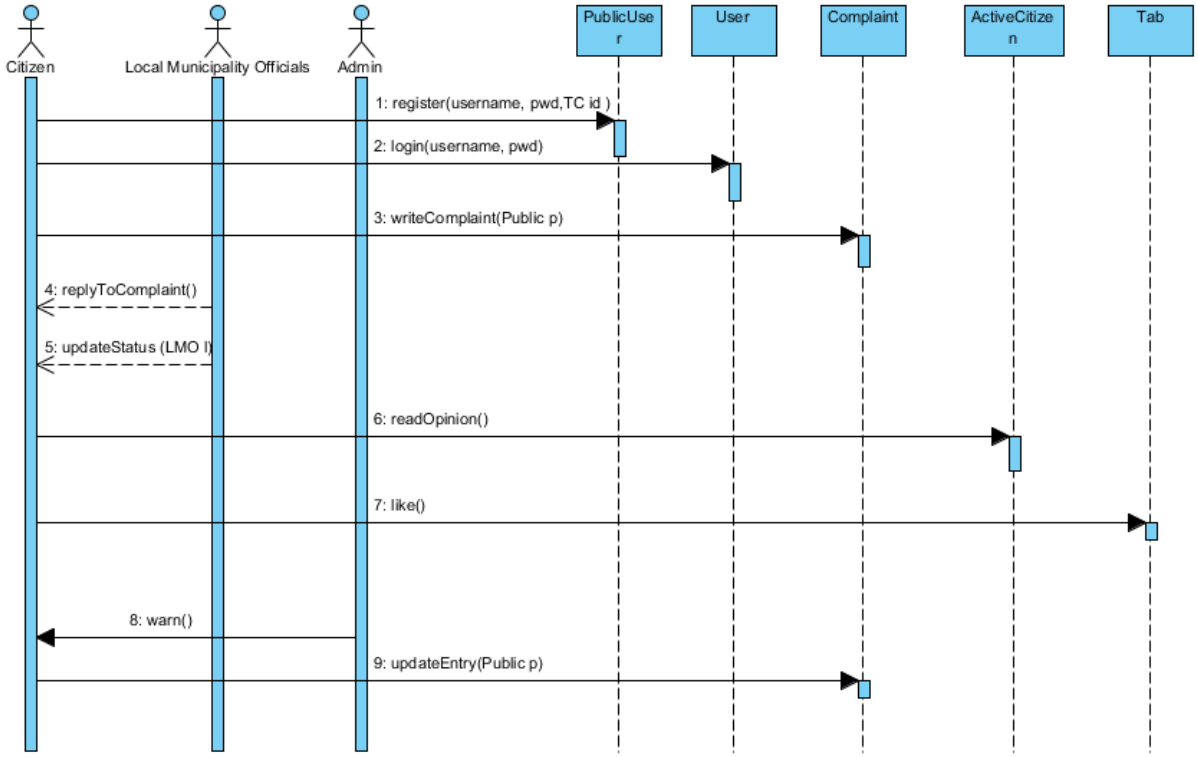

Figure 14: Sequence diagram for a citizen

The state diagram of the system describes the status of the object and the events that trigger the status change. In Figure 15, the state diagram shows the status change of the complaint that is submitted to the system. When official view complaints as a list but do not perform any operation state will be pending again. If official open complaint/suggestion page by clicking Evaluate button, state will become Being Evaluated. Then, official can accept or reject. If official do not take action like accept or reject state will not change and stay Being Evaluated.

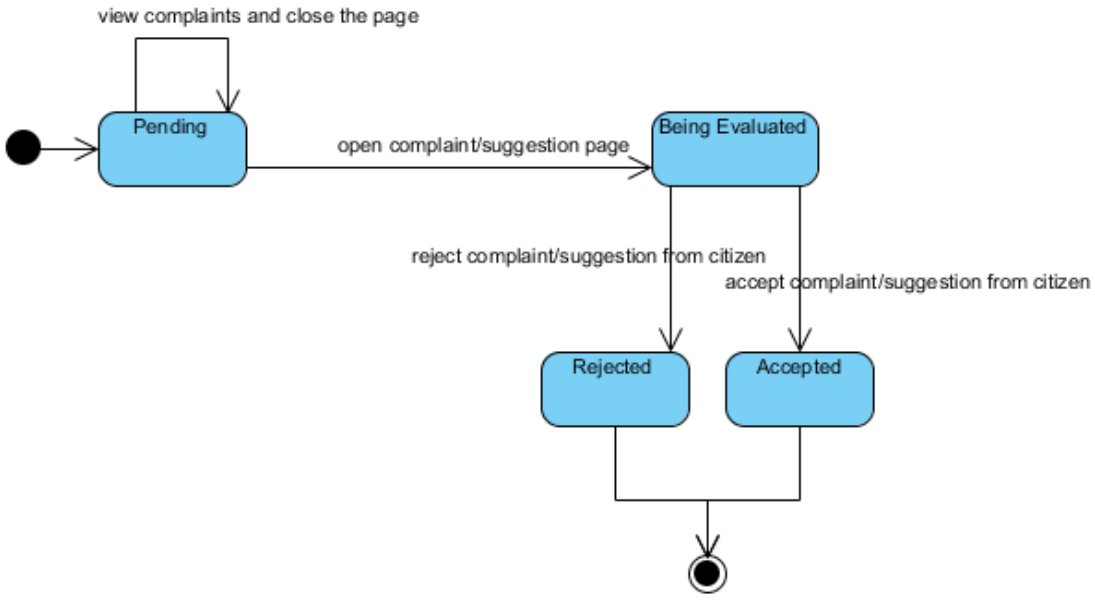

Figure 15: State diagram for the complaint/suggestion of the citizen

5. Technical Problem Analysis

The technical problems are described in a way that, first the initial state and then the desired state is described. In our system, there is no initial state because there is not an existing system that is similar to our system. Therefore, the descriptions of the problems given below explain the desired states for each problem.

P1. How to authenticate

P1.1 How to isolate different user groups

There are various user groups in the system with different privileges. These groups should be well-isolated and privileges should not be interlaced. For example, a citizen should not delete a different citizen's comment.

P1.2 How to protect an account

User enters the system with his/her username and password. This password is so strong that hacking them is extremely difficult.

P1.3 How to store passwords in the database

User passwords are encrypted in the database of the system because admin cannot see passwords and use any citizen's account.

P2. How to store pictures

Citizens upload pictures to the system together with their complaints/suggestions. These pictures should be efficiently stored in the system.

P3. How to store comments/complaints/suggestions and their states persistently

Entries of users should be persistently stored in the database since many users enter various entries such as comments/complaints/suggestions under different subtitles and these should be visible every time.

P4. How to specify state of entries

Entries entered by citizens should be confirmed for not abusing system before publishing. Hence, entries have different states in controlling process. Citizens can also follow the state of their entries. These should be well specified by application domain experts.

P5. How to make connection secure

Since connection between mobile device and the server is wireless someone can sniff these packages and get information. Hence this connection must be encrypted.

P6. How to provide compatibility for different operating systems

Application should also work for mobile devices that use operating systems like iOS, windows phone 7, besides Android OS. Since there are different types of mobile devices using different operating systems, the system should serve to those users' needs as well.

P7. How to make the system resistant to upgrade in Android OS

Application will be developed according to the latest version of Android OS. However, the operating systems are upgraded frequently and the system should make sure that it works for different versions of the operating system.

P8. How to provide compatibility for different devices

Application will be developed for mobile phones with Android OS. Since there are different types of mobile devices using Android OS, the system should be compatible for these devices.

P9. How to present new places

Citizens can enter new places like restaurants, cinemas, cafes and shopping centers. Entering new places is done based on a specific template. Which components this template includes is specified by application domain experts.

P10. How to make user-friendly interfaces

The system is for all citizens from different backgrounds. Therefore, the system should be easy to use by citizens and consequently its interfaces should be user friendly.

P11. How to provide system recovery in case of server crash

The data of complaints/suggestions and their states are stored in the server side and there must be a recovery mechanism in case the server crashes.

P12. How to make the system responsive 7 days 24 hours

One of the advantages of the system is to remove the unavailability of the service provided by the municipalities, since their accessibility are limited to working hours.

P13. How to serve multiple citizens at the same time

This is a concurrency issue, and the system must be responsive to the multiple users without performance loss or any transaction problem. Data should be managed efficiently and correctly while multiple users use the system at the same time.

6. Domain Analysis

Domain of the project will be decided according to the current needs in a typical municipality.

6.1. Domain & Feature Model

Solution domain is described in order to determine the domains to be analyzed to solve the technical problems. Therefore, the solution domain is derived from the technical problems and defined in section 5 and included in Table 1.

SOLUTION DOMAIN
Connectivity
Storage of data
Concurrency
User Interface
Recovery
Consistency
System Management
Sustainability
Security
Acquirement
Production Management

Table 1: Solution Domain

Each solution domain is mapped to a technical problem, for instance authentication problem can be solved by analyzing the security domain and the specification of the state of entries can be solved by deep analysis of consistency domain. All problems and the corresponding solution domains with the associated priorities are given in Table 2. Highest priority is shown as 1, greater number shows decreasing priority.

Problem	Solution Domain	Priority
How to authenticate	Security Connectivity Acquirement System Management	1 2 2 2
How to store pictures	Storage of data Acquirement	1 1
How to store comments/complaints/suggestions and their states persistently	Storage of data Acquirement	1 1
How to specify state of entries	Storage of data Acquirement Consistency	1 1 2
How to make connection secure	Security	1
How to provide compatibility for different devices	Production Management Sustainability	2 1
How to provide compatibility for different operating systems	Production Management Sustainability	2 1
How to make the system resistant to upgrade in Android OS	Production Management Sustainability	2 1
How to present new places	User Interface	2
How to make user-friendly interfaces	User Interface	1
How to provide system recovery in case of server crash	Recovery Storage of data Consistency System Management	1 1 1 1
How to make the system responsive 7 days 24 hours.	Concurrency System Management	1 1
How to serve multiple citizens at the same time	Concurrency Consistency	1 1

Table 2: Mapping Solution Domains to Problems

The system includes different categories which the user will be able to choose and make complaints/suggestions or write comments about. The system will be customized according to the user needs, i.e. the city administrators will be able to choose the categories and the properties that they want to see in their city municipalities. The feature diagram in Figure 16 allows the system to be customized according to the user needs by allowing some categories to be optional.

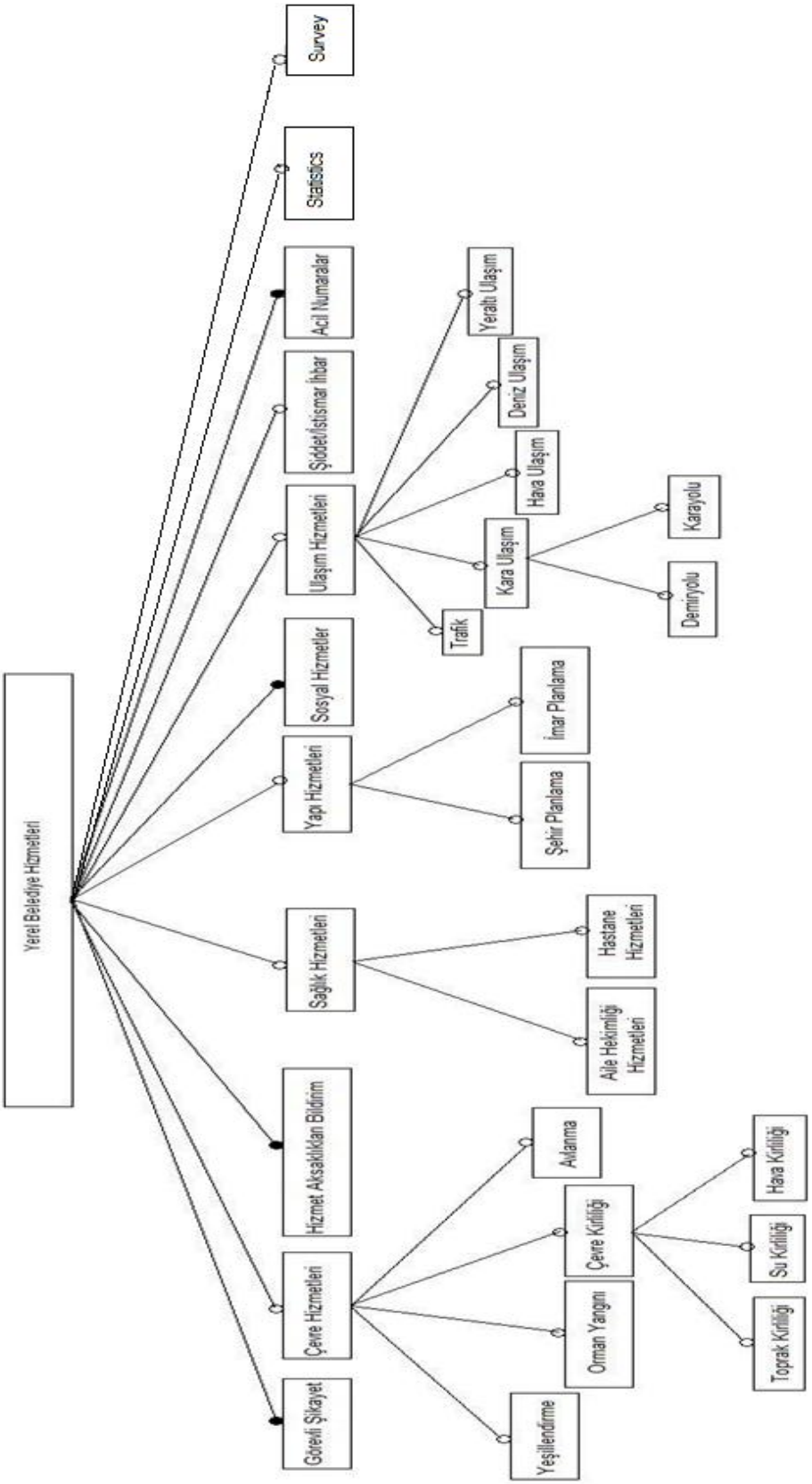

Figure 16: Feature Model

6.2.Knowledge Sources

There are different knowledge sources and each of them can be analyzed in different forms, an expert, an application or a textbook can be used. In Table 3, the knowledge sources and the corresponding forms are shown for overall solution domain.

ID	Knowledge Source	Form
KS1	Computer Networking [Kurose & Ross 2010]	Textbook
KS2	Database Management Systems [Ramakrishnan & Gehrke 2003]	Textbook
KS3	Concurrency Control and Recovery in Database Systems [Bernstein et al. 1987]	E-book
KS4	HUMAN-COMPUTER INTERACTION [Dix et al. 2004]	E-book
KS5	System Manager	Person
KS6	Local Municipality Websites – Fatih Mobil	Application
KS7	Wireless and Mobile Network Architectures [Imrich Chlamtac 2001]	Textbook
KS8	Information Storage and Management [G. Somasundaram, Alok Shrivastava 2009]	Textbook
KS9	Advanced Android Development [Mark Murphy 2011]	Textbook
KS10	Database security [Fuqini, Castano, Martello 1994]	Textbook
KS11	Production Management: Making Shows Happen [Peter Dean 2002]	Textbook
KS12	Production Manager- Innova	Person
KS13	Hello, Android: Introducing Google's Mobile Development Platform [Ed Burnette 2010]	Textbook

Table 3: Knowledge Sources

In Table 4, the knowledge sources and the corresponding forms are shown for connectivity solution domain.

ID	Knowledge Source	Form
KS1	Computer Networking [Kurose & Ross 2010]	Textbook
KS2	Database Management Systems [Ramakrishnan & Gehrke 2003]	Textbook
KS7	Wireless and Mobile Network Architectures [Imrich Chlamtac 2001]	Textbook
KS8	Information Storage and Management [G. Somasundaram, Alok Shrivastava 2009]	Textbook

Table 4: Knowledge Sources for connectivity domain

In Table 5, the knowledge sources and the corresponding forms are shown for consistency solution domain.

ID	Knowledge Source	Form
KS1	Computer Networking [Kurose & Ross 2010]	Textbook
KS2	Database Management Systems [Ramakrishnan & Gehrke 2003]	Textbook
KS7	Wireless and Mobile Network Architectures [Imrich Chlamtac 2001]	Textbook

Table 5 Knowledge Sources for consistency domain

In Table 6, the knowledge sources and the corresponding forms are shown for User Interface solution domain.

ID	Knowledge Source	Form
KS4	HUMAN-COMPUTER INTERACTION [Dix et al. 2004]	E-book
KS6	Local Municipality Websites – Fatih Mobil	Application
KS9	Advanced Android Development [Mark Murphy 2011]	Textbook
KS13	Hello, Android: Introducing Google's Mobile Development Platform [Ed Burnette 2010]	Textbook

Table 6 Knowledge Sources for user interface domain

In Table 7, the knowledge sources and the corresponding forms are shown for Storage of data solution domain.

ID	Knowledge Source	Form
KS2	Database Management Systems [Ramakrishnan & Gehrke 2003]	Textbook
KS4	HUMAN-COMPUTER INTERACTION [Dix et al. 2004]	E-book
KS8	Information Storage and Management [G. Somasundaram, Alok Shrivastava 2009]	Textbook
KS9	Advanced Android Development [Mark Murphy 2011]	Textbook
KS13	Hello, Android: Introducing Google's Mobile Development Platform [Ed Burnette 2010]	Textbook

Table 7 Knowledge Sources for storage of data domain

In Table 8, the knowledge sources and the corresponding forms are shown for acquirement solution domain.

ID	Knowledge Source	Form
KS2	Database Management Systems [Ramakrishnan & Gehrke 2003]	Textbook
KS8	Information Storage and Management [G. Somasundaram, Alok Shrivastava 2009]	Textbook
KS9	Advanced Android Development [Mark Murphy 2011]	Textbook
KS13	Hello, Android: Introducing Google's Mobile Development Platform [Ed Burnette 2010]	Textbook

Table 8 Knowledge Sources for acquirement domain

In Table 9, the knowledge sources and the corresponding forms are shown for concurrency and recovery solution domain.

ID	Knowledge Source	Form
KS3	Concurrency Control and Recovery in Database Systems [Bernstein et al. 1987]	E-book

Table 9 Knowledge Sources for concurrency and recovery domain

In Table 10, the knowledge sources and the corresponding forms are shown for system management solution domain.

ID	Knowledge Source	Form
KS5	System Manager	Person
KS9	Advanced Android Development [Mark Murphy 2011]	Textbook

Table 10 Knowledge Sources for system management domain

In Table 11, the knowledge sources and the corresponding forms are shown for production management solution domain.

ID	Knowledge Source	Form
KS11	Production Management: Making Shows Happen [Peter Dean 2002]	Textbook
KS12	Production Manager- Innova	Person

Table 11 Knowledge Sources for production management domain

6.3.Derived Concepts

In Table 12, the solution domain concepts for each solution domain are listed.

Solution Domain	Solution Domain Concept
Connectivity	Webserver Manager, Network Manager
Storage of data	Data Manager
Concurrency	Data Manager, Concurrency Manager
User Interface	UI Manager
Recovery	Data Manager, Recovery Manager
Consistency	Data Manager, Recovery Manager, Concurrency Manager
System Management	System Manager
Security	Security Manager, Network Manager, Webserver Manager
Acquirement	Data Manager
Production Management	Adaptability Manager

Table 12: Solution Domain Concepts

The sub-level conceptual architecture for the solution domains such as storage of data, concurrency, recovery, consistency, and acquirement is illustrated below in Figure 17.

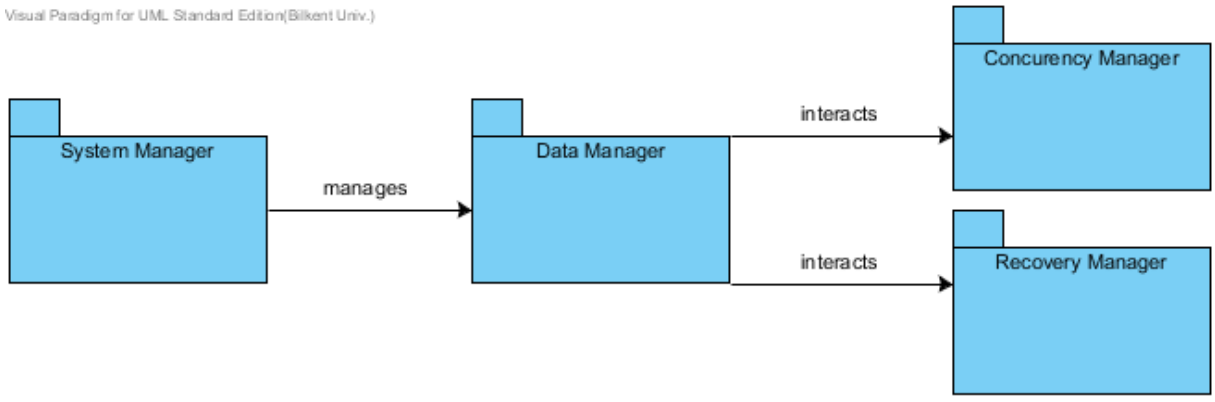

Figure 17: Sub-Level Conceptual Architecture

The concepts in the sub-level conceptual architecture in Figure 17 are explained in detail in Table 13.

Concept	Description of the Solution Domain Concept
System Manager	System manager is responsible for management of the system.
Data Manager	Database manager is responsible for creation, maintenance and the use of data in the system
Concurrency Manager	Concurrency manager is a manager that ensures many users can access the data at the same time so that data transactions are performed concurrently.
Recovery Manager	Recovery management is a management that saves the data from failure, damage, and corruption when data cannot be accessed

Table 13: Description of Solution Domain Concepts in Figure 18

The sub-level conceptual architecture for the solution domains such as connectivity, security is illustrated below in Figure 18.

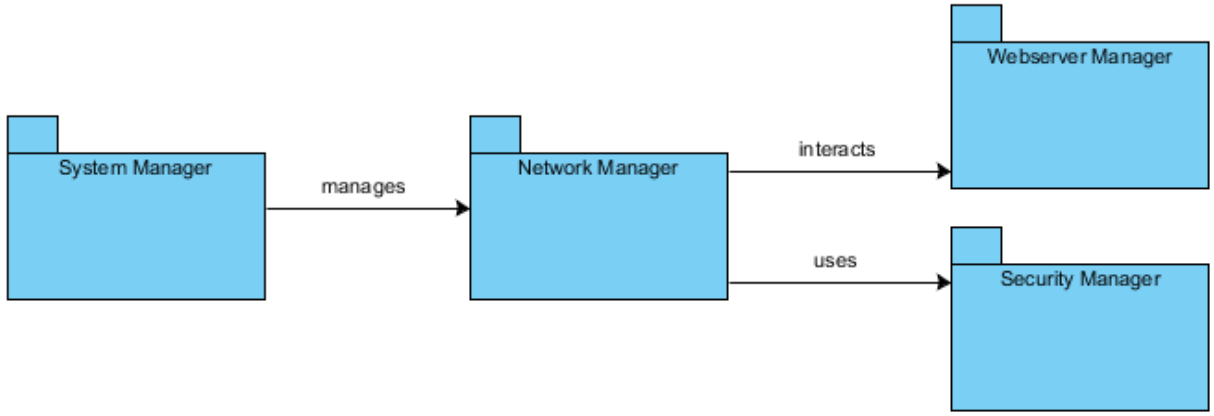

Figure 18: Sub-Level Conceptual Architecture

The concepts in the sub-level conceptual architecture in Figure 18 are explained in detail in Table 14.

Concept	Description of the Solution Domain Concept
Network Manager	Network manager is a management that is responsible for administration and management of network of the system.
Web Server Manager	Web Server management deals with management of all aspects of web services.
Security Manager	Security management is a management for ensuring network is protected from unauthorized users.

Table 14: Description of Solution Domain Concepts in Figure 19

7. Top-level Context Diagram

Context diagram describes the general, i.e. external and internal, relationships of the sub systems and in Figure 19, the context diagram of our system is included. The system is named Active Citizen and there are two sub systems inside it, database and network which interact with the external entities which are the users of the system.

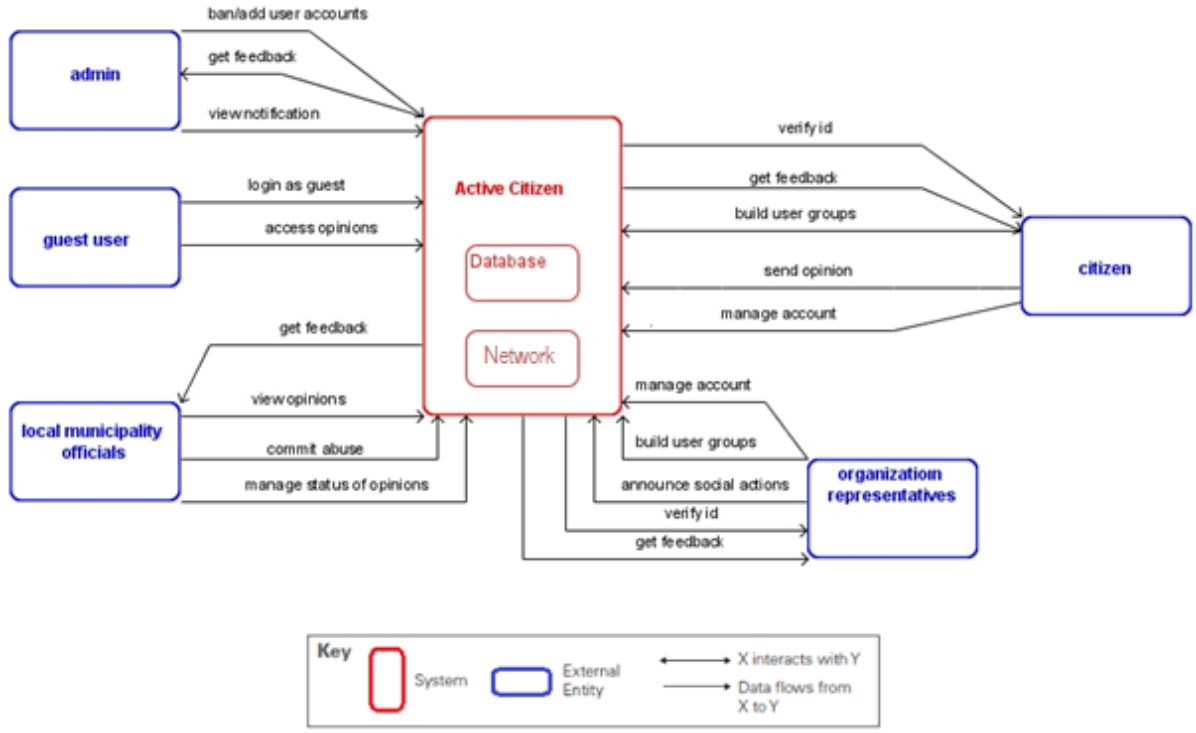

Figure 19: Top-level Context Diagram

8. Software Architecture Design

The conceptual software architecture of the system including the internal relations of the domain concepts is given in Figure 20.

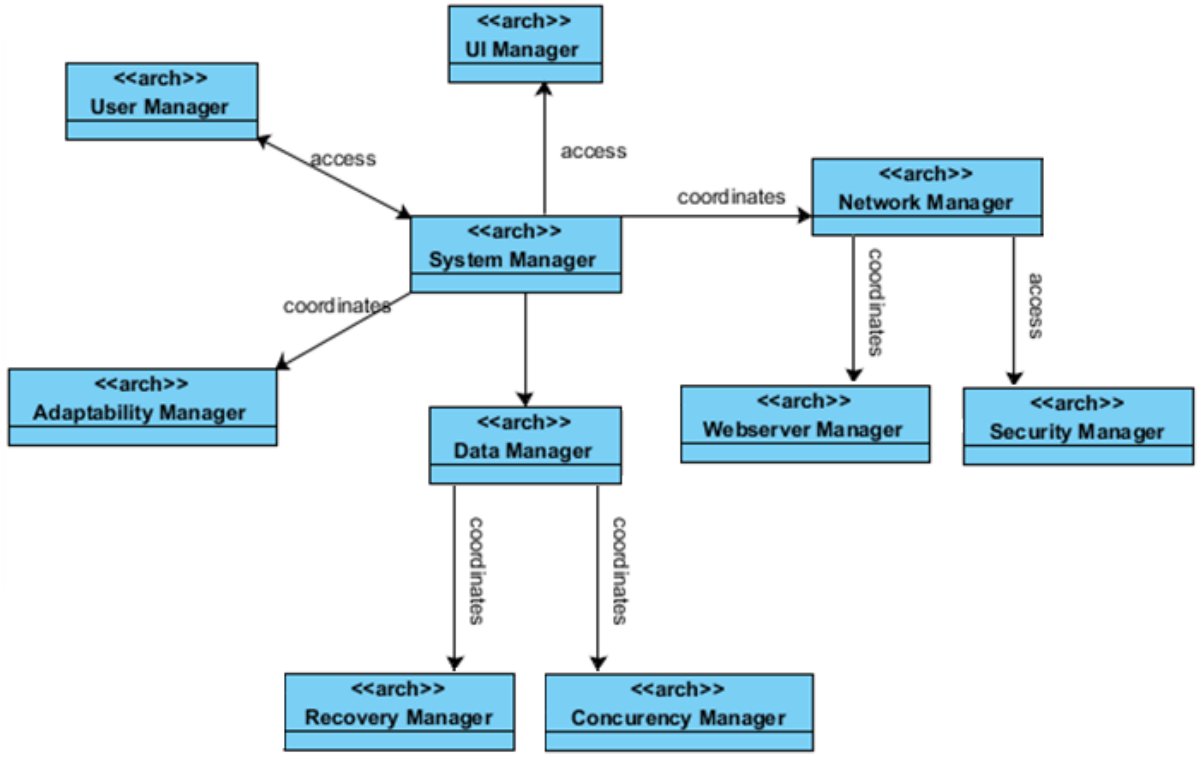

Figure 20: Conceptual Software Architecture

Figure 20 shows the top level structure of the conceptual software architecture. System manager is the main manager of the system; it has access to UI manager, user manager and data manager which coordinates recovery and concurrency manager. System manager also coordinates adaptability and network manager that coordinates web server and security manager. The role of the managers of the system is explained in Table 13 and 14.

9. Module Views

9.1.Decomposition View

Decomposition view is one of the module views, which shows the modules in the system. The decomposition relation can be defined as the specialization of "is-part-of" relation between modules. The top-level decomposition view of the system is given in Figure 21. Active Citizen consists of two basic modules which are WorkFlowManager and ProcessManager. ProcessManager is the citizen side of the system which includes subsystems within it. The

subsystems provide the data, network, user interface and the initial configuration of the system for different municipalities which is done by Municipality Management. WorkFlowManager conducts the municipality side work flow and the corresponding feedbacks towards the user from the municipality side.

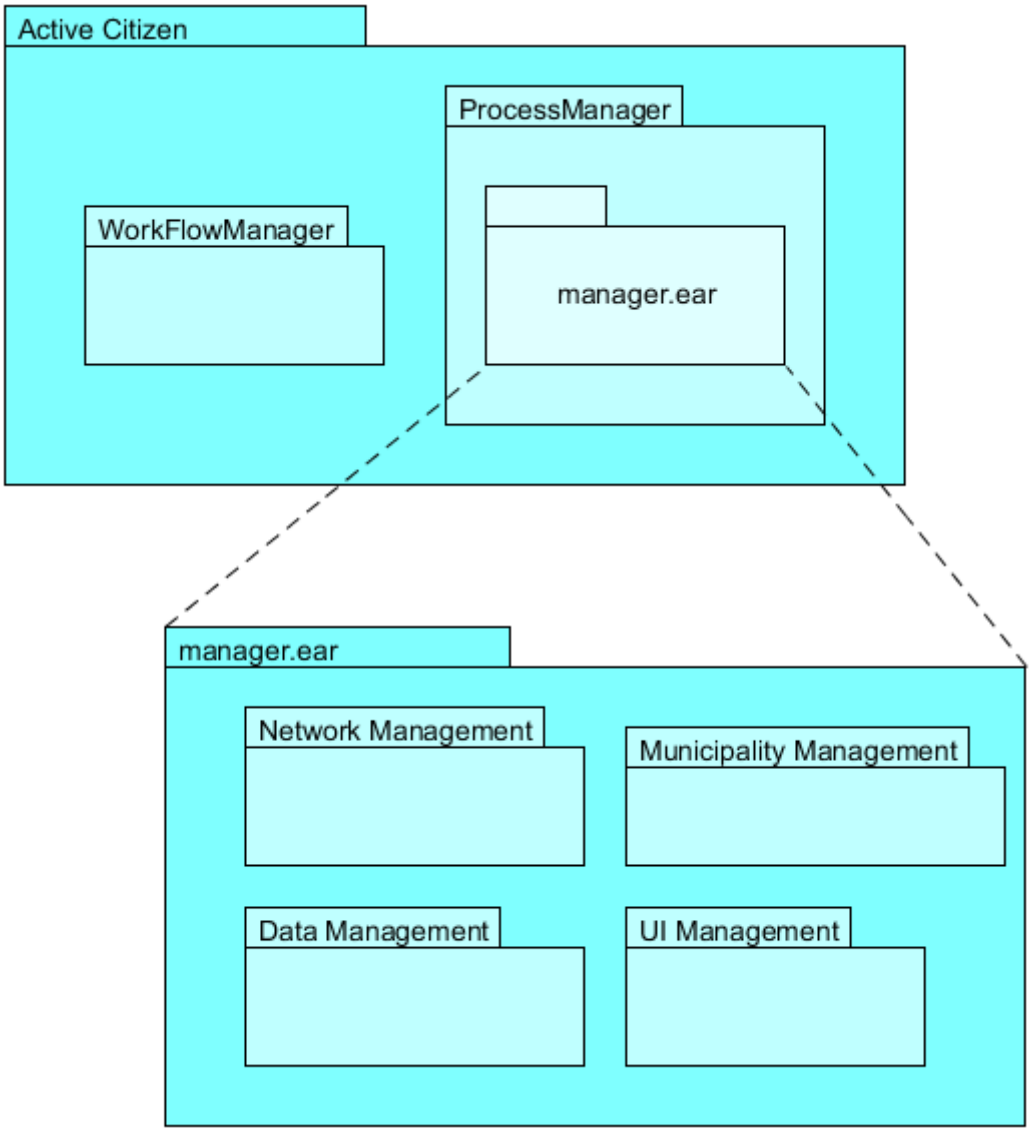

Figure 21: Top-level decomposition view of the system

The ProcessManager subsystem consists of 4 modules which are Data Management, Municipality Management, Network Management and UI Management. The decomposition view for the subsystems of these packages is shown in Figure 22. Municipality Management provides the configuration of the application according to different municipalities. The module includes Category Provider in order for the municipalities to choose from the feature diagram. The categories are adapted to the system with Module adapter subsystem and MapDomain subsystem provides the categories chosen from the domain to be mapped to the application.

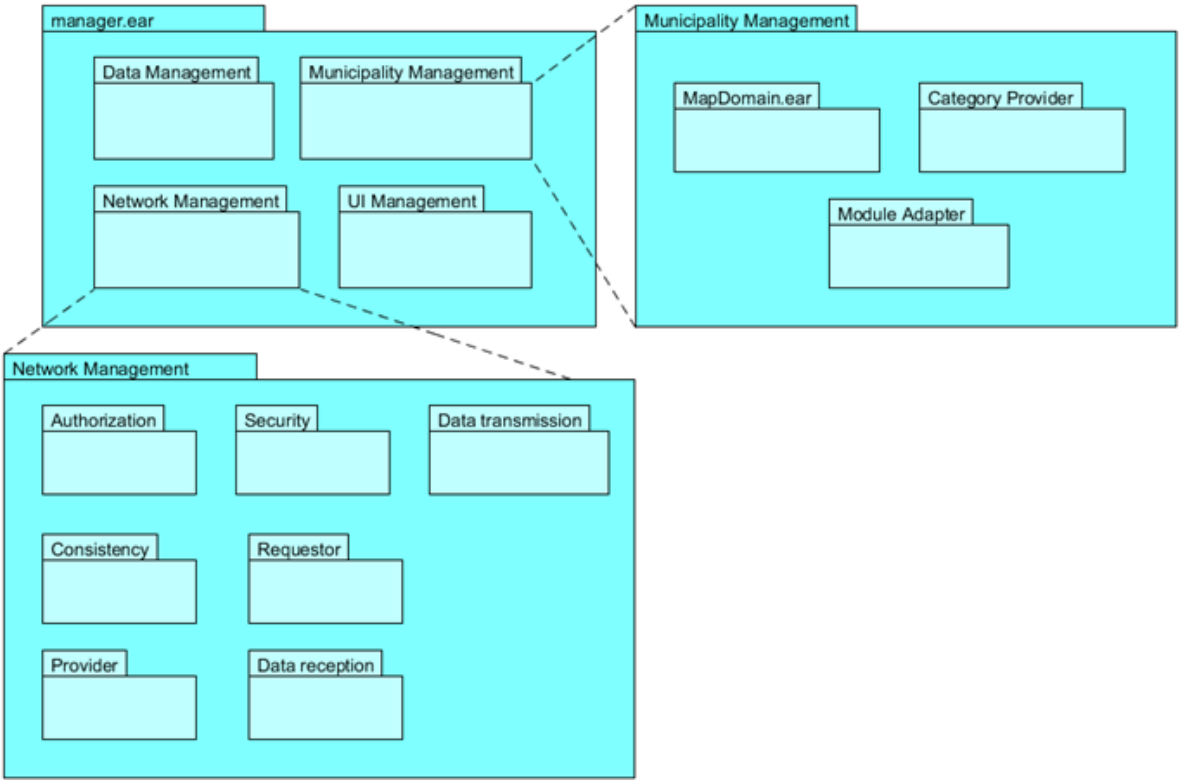

Figure 22: Subsystem decomposition of manager.ear

The Data Management subsystem which is part of the manager.ear consists of 8 modules each of which provides the data management for different parts of the system. The decomposition view for the Data Management subsystem is shown in Figure 23. Backup module provides the data recovery in case of any server crash. Account module includes the registration and login information for different users of the system. Status, Image and Textual Content Handling modules provide the storage of the status of the complaints/suggestions of the citizens, the images that they submit to the system, and any textual entry written to the system subsequently. Citizen, Local Municipality Official and Nongovernmental Organization modules store the additional data related to the authorization of the users, such as group information for citizens.

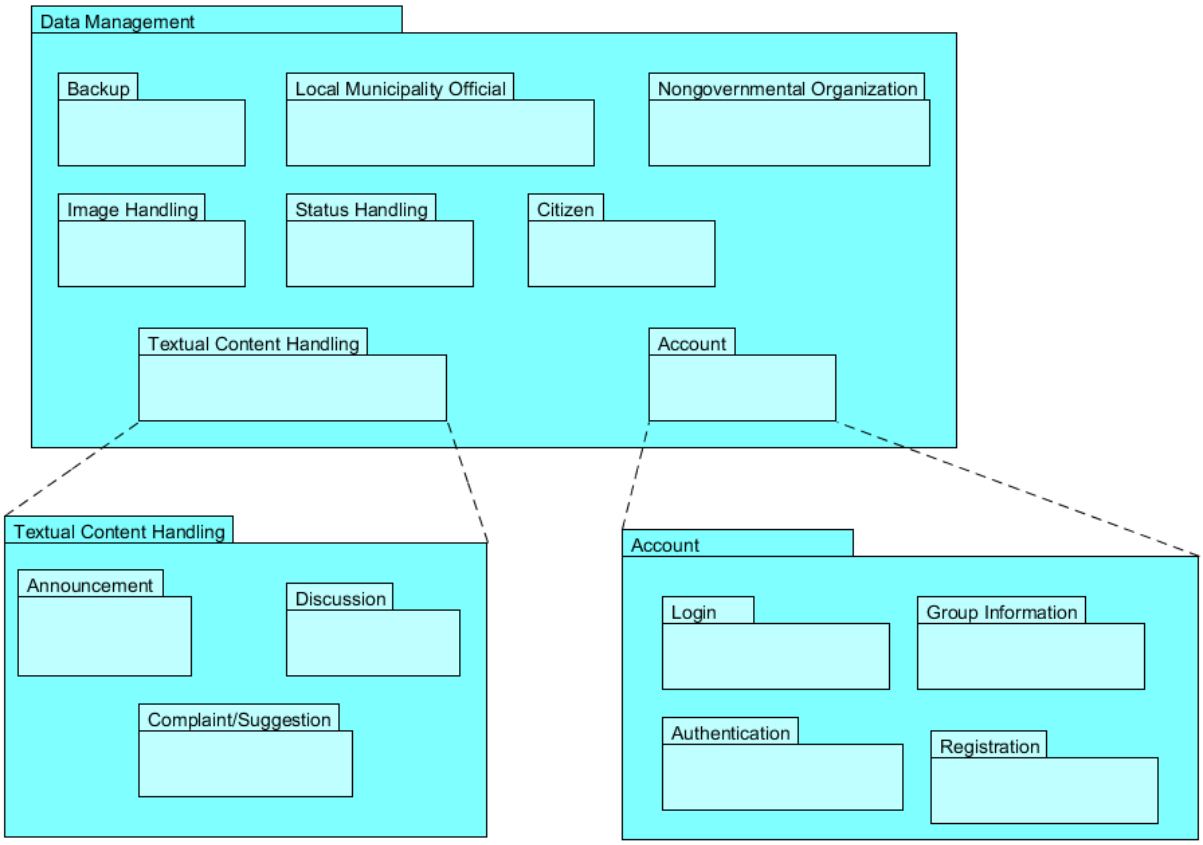

Figure 23: Subsystem decomposition of Data Management

The Network Management subsystem which is part of the manager.ear consists of 7 modules which provide the data transmission over the network in a secure, consistent way within authorization limits. The decomposition view for the Network Management subsystem is shown in Figure 24. Authorization module provides the users to access the data that they have authorization to see. Consistency module provides the data to be transmitted over the network in a consistent manner. Security module guarantees that the information being transmitted on the network is secure. Data Transmission and Data Reception modules allow the image, textual data, status of complaint/comments and account information to be transmitted to server or received from the server subsequently.

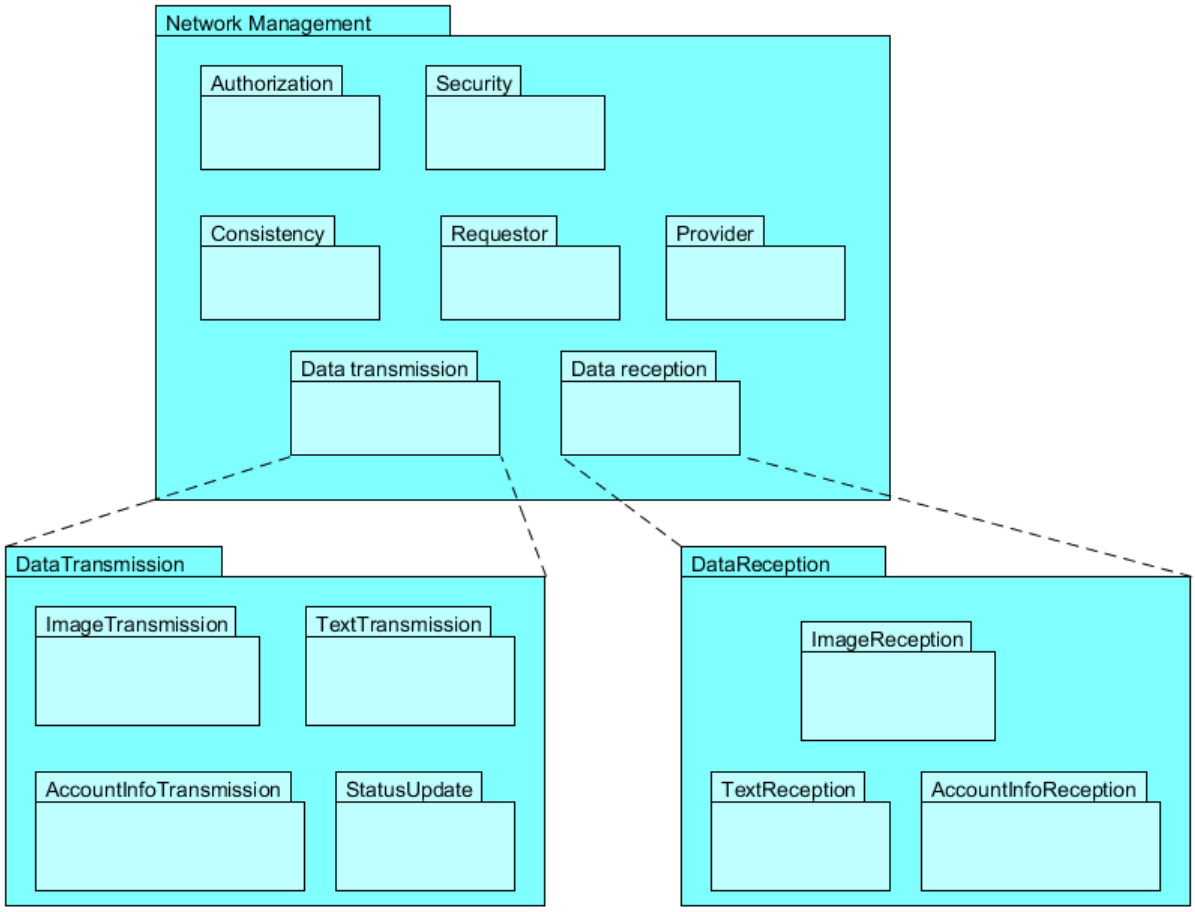

Figure 24: Subsystem decomposition of Network Management

9.2.Context Diagram for Decomposition View

The context diagram for decomposition view depicts the packages' "is-part-of" relations in a larger system. The larger system consists of the other systems and "Active Citizen" system. The context diagram for decomposition view is shown in Figure 25. Active citizen is shown as nested inside the larger system which is Municipality Management System. There are also 3 existing systems in the whole system and they serve to municipality side whereas our application responds to citizens requests. Municipality Mayor Service Tracking System is used as an Android application and Mayor can trace the status of the complaints coming from citizens in terms of small districts in municipality. Municipality workflow management system is used in order to keep track of the workflow from the point that the complaint arrives until it is solved. Municipality Field Tracking System is an Android application used by the field officials while they conduct their tasks in cooperation.

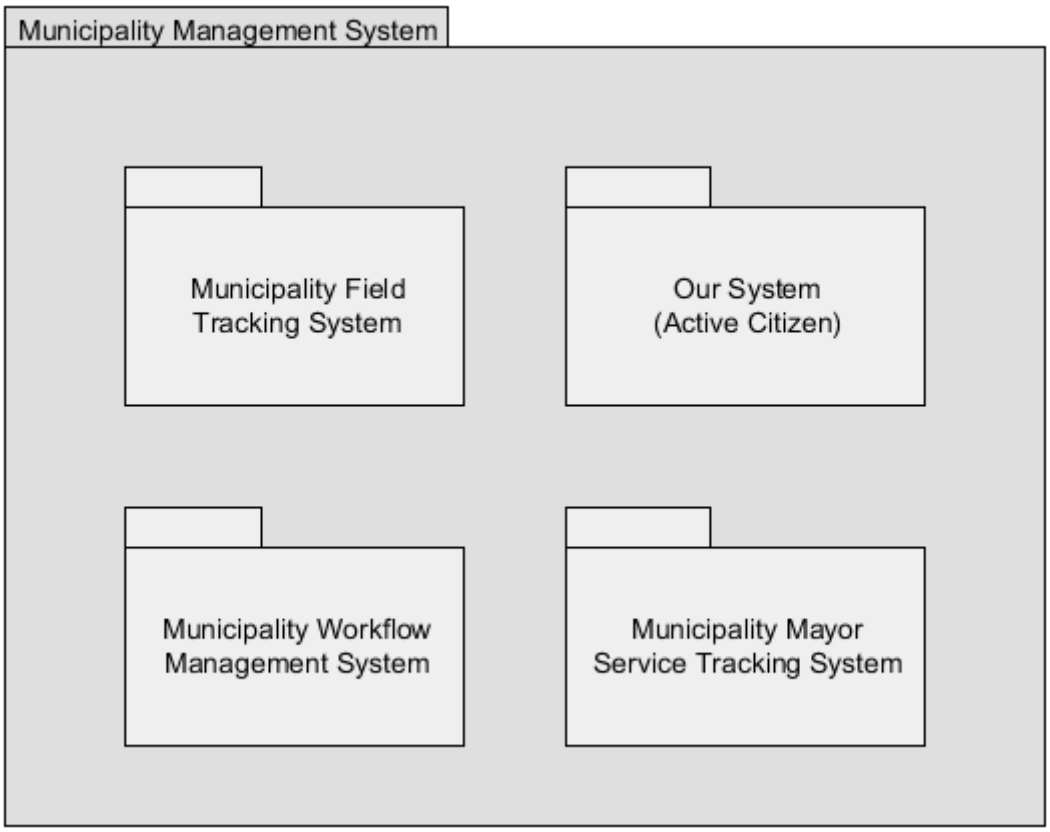

Figure 25: Context Diagram for Decomposition View

9.3.Uses View

Uses view is one of the module views, which shows the "uses" relation between the modules in the system. Uses view is used for planning incremental development, system extensions and subsets, debugging and testing, and gauging the effects of specific changes. The relation of which module depends on the correct functioning of which module is shown on the uses view. The uses view for the system in general and for the manager.ear is shown in Figure 26. The uses view for Data Management and Network Management modules is given in Figure 27. In order to provide authorization in the Network Management module, Account module in Data Management should be used, because the authorization for data access is decided according to the account information for the users.

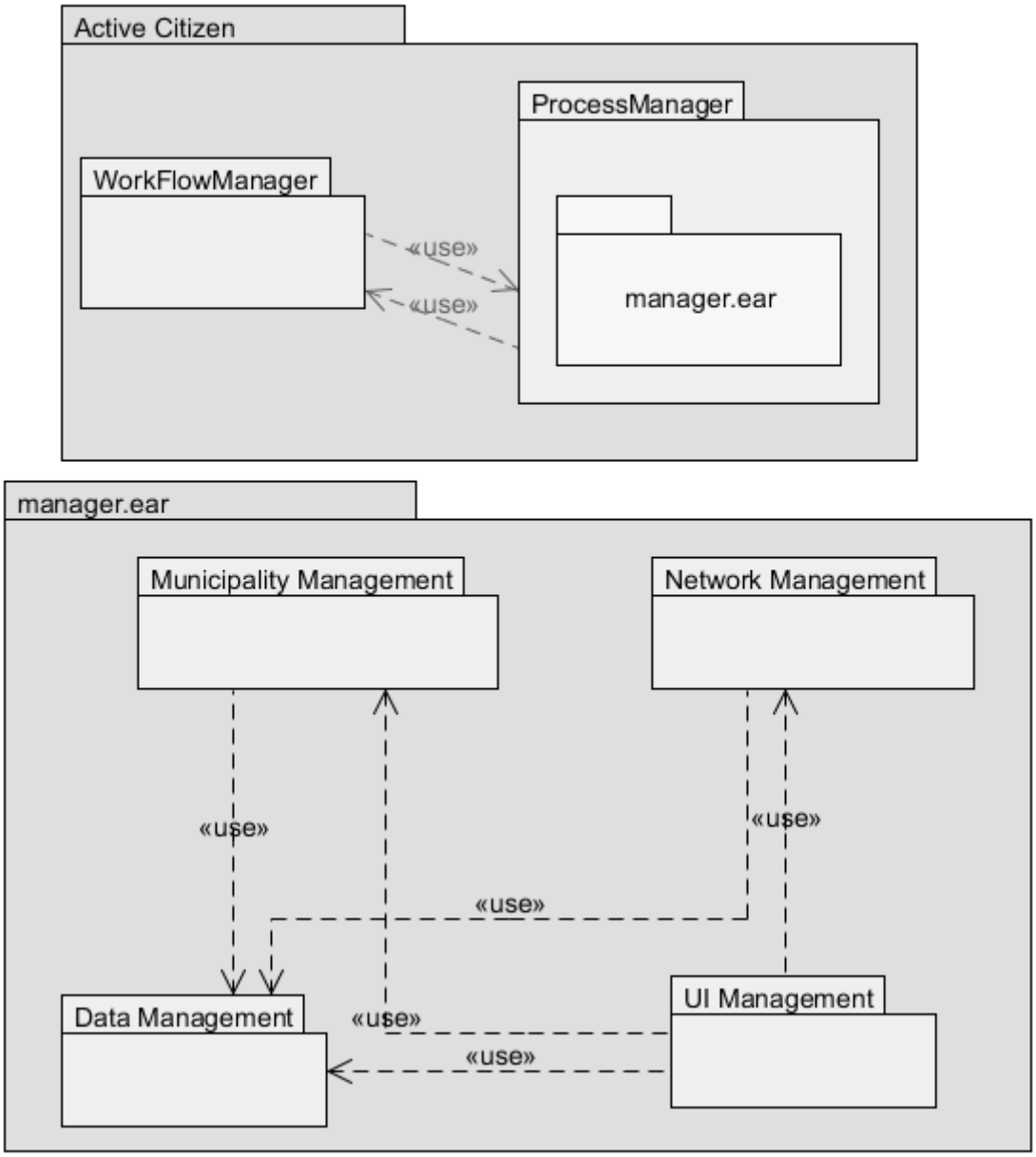

Figure 26: Uses view for the subsystems of manager.ear

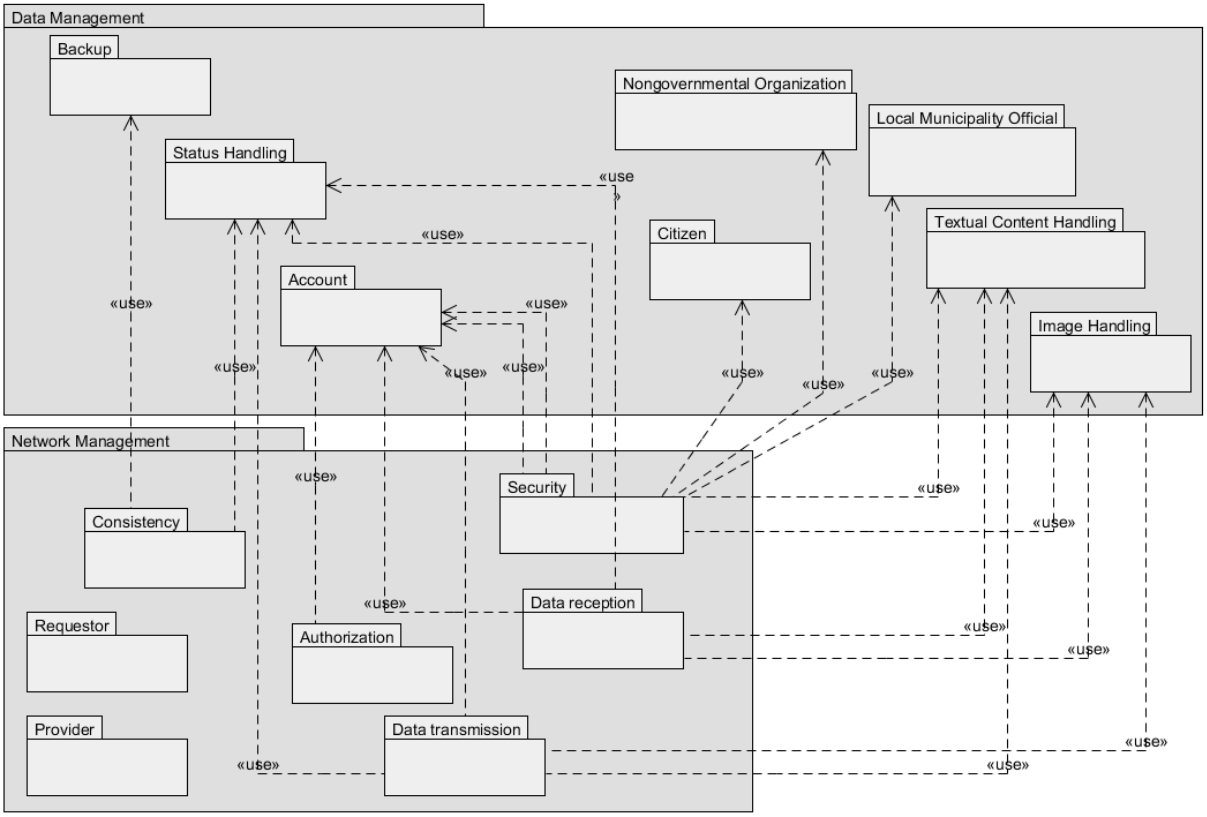

Figure 27: Uses view for Data Management and Network Management

For the Municipality Management subsystem, the uses view is shown in Figure 28. In order to map the domains to the system, i.e. to provide configuration of the system for different municipalities, MapDomain.ear uses Module adapter and category Provider modules.

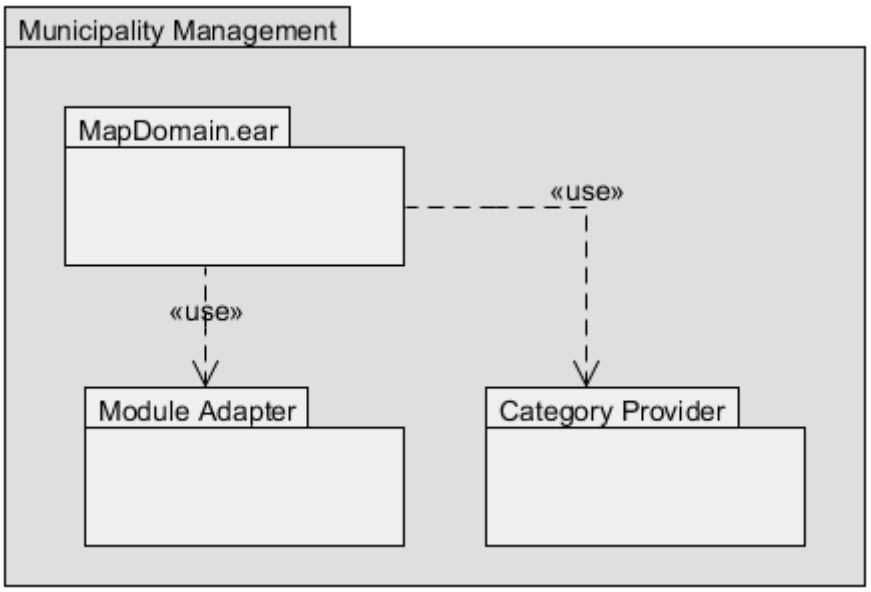

Figure 28: Uses view for Municipality Management

9.4.Context Diagram for Uses View

The context diagram for uses view depicts which external systems use Active Citizen or used by Active Citizen. The context diagram for uses view is shown in Figure 29. Municipality Mayor Service Tracking System uses Active Citizen Application in order to feed the complaints and ratings of the municipalities. As mentioned before, by this way mayor can trace the status of the complaints coming from citizens in terms of small districts in municipality. Municipality workflow management system uses Active citizen in order to get the complaints to the system and is used by Active Citizen to update the status of the complaint on the citizen side. Municipality Field Tracking System is an Android application that uses Active Citizen in order to feed data related to complaints as well.

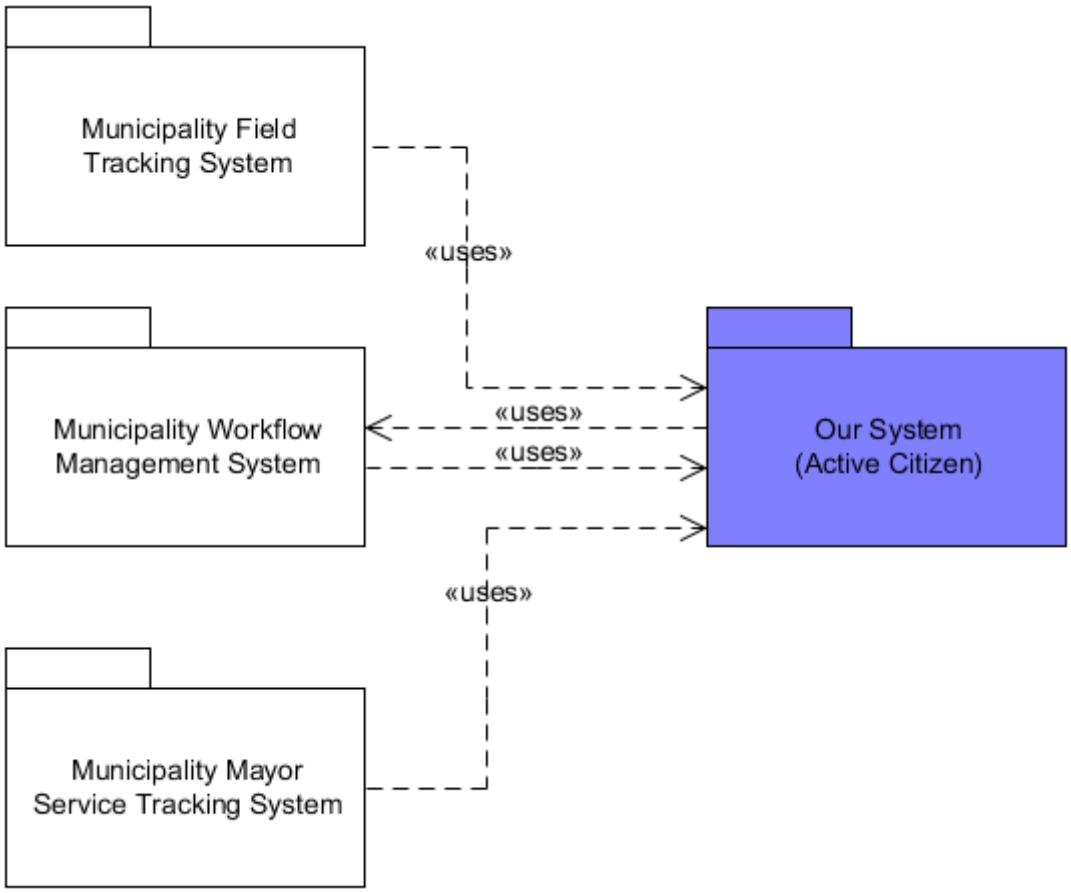

Figure 29: Context Diagram for Uses View

9.5.Generalization View

Generalization view is one of the module views, which shows “is-a” relation among modules in the system. Generalization view shows the extension and evolution, local changes or variations, commonalities between the modules and production of incremental descriptions in

the system. Generalization view allows the reusability of the system. The generalization view of Active Citizen is given in Figure 30.
The observer pattern can be detected at this point and the updates to the data, i.e. write and update in the database server will be observed by the users and the changes will be reflected on the users' side and UIManager.

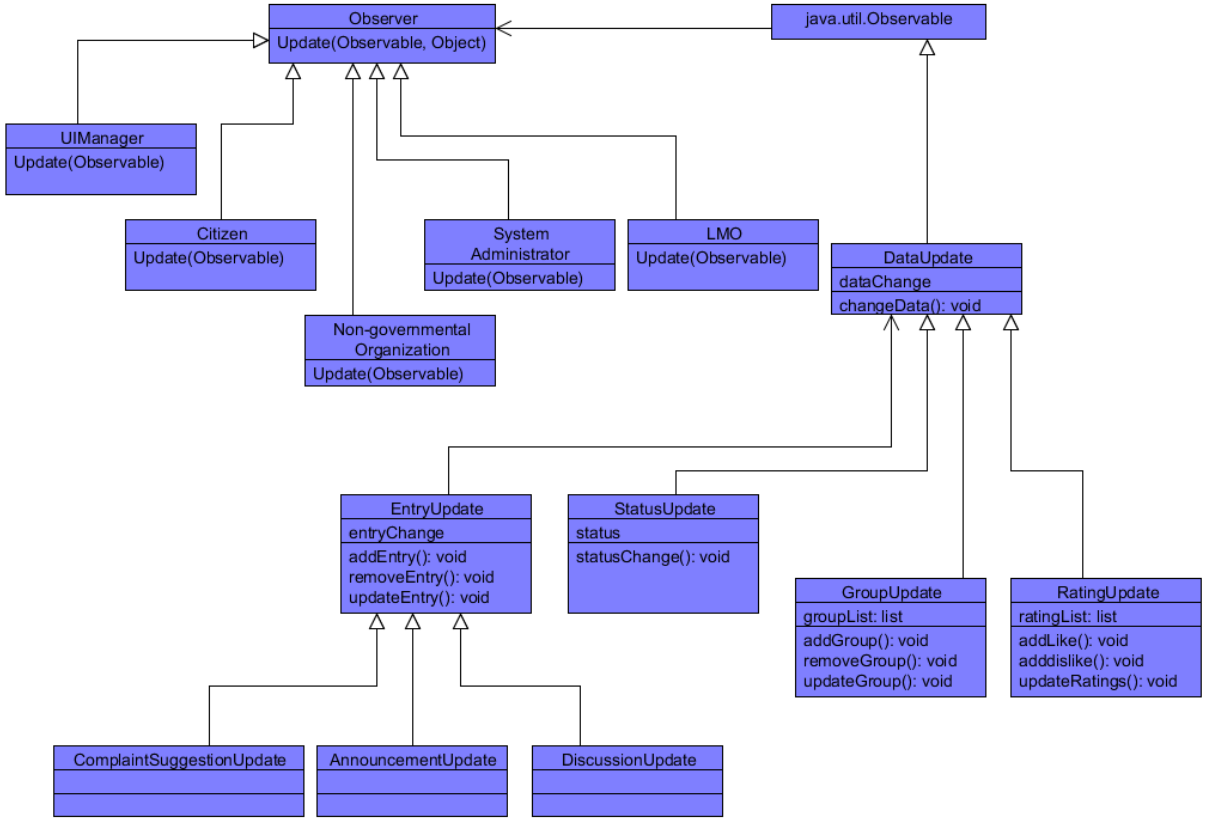

Figure 30: Generalization view of the system

9.6.Layers view

Our system uses three layered structure which encompasses presentation layer, application layer, business logic layer and data access layer as shown in Figure 31.

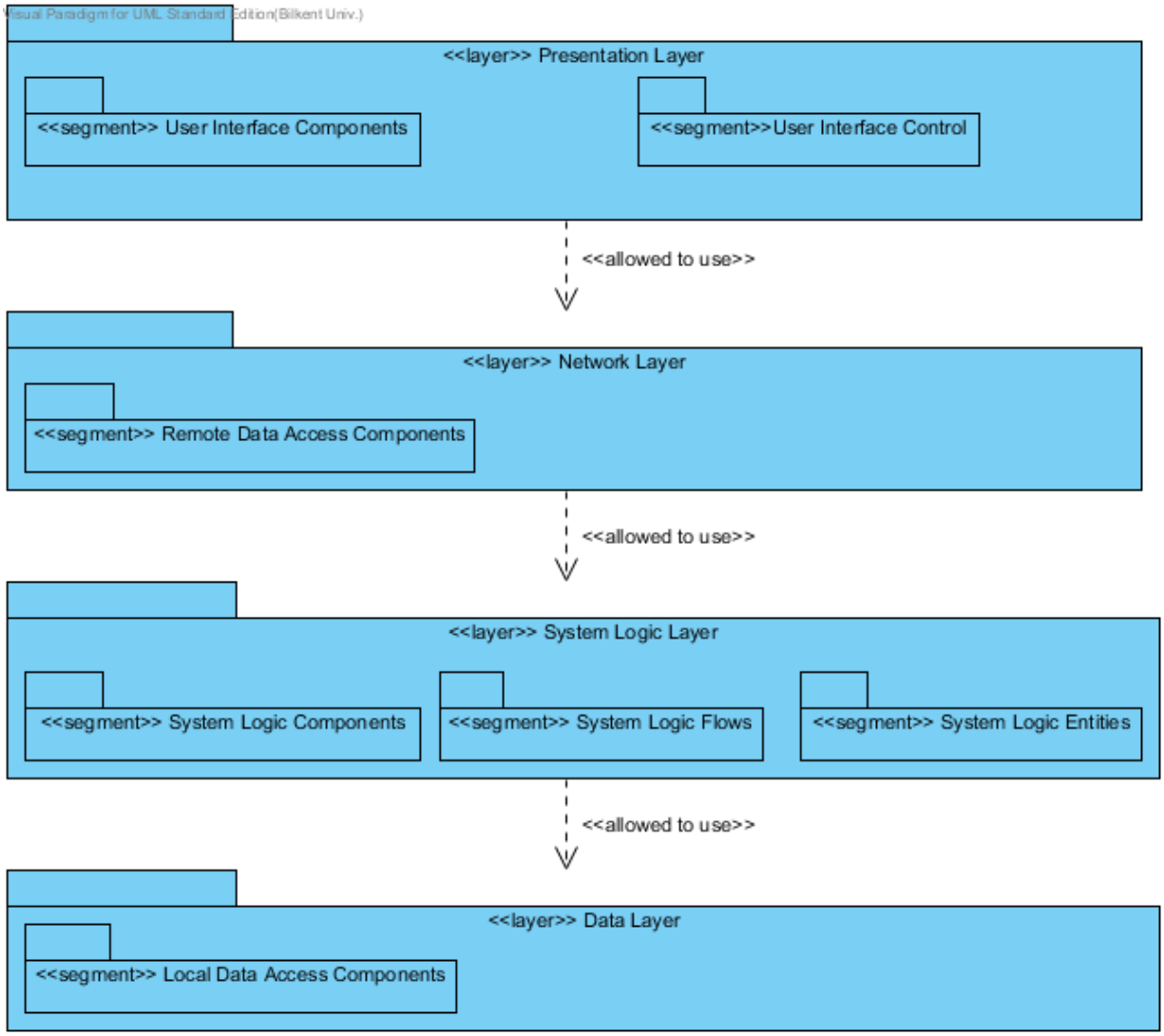
Visual Paradigm for UML Standard Edition(Bilkent Univ.)
<<layer>> Presentation Layer
<> User Interface Components
<>User Interface Control
<<allowed to use>>
<<layer>> Network Layer
<> Remote Data Access Components
<<allowed to use>>
<<layer>> System Logic Layer
<> System Logic Components
<> System Logic Flows
<> System Logic Entities
<<allowed to use>>
<<layer>> Data Layer
<> Local Data Access Components

Figure 31: Layered view of the system [3]

Presentation Layer: This layer is responsible for representation of data to the user. This layer includes UI components that provide interaction with the user of the system. Presentation layer also deals with displaying and managing which data is displayed to the user. This layer uses network layer for remote data access.

Network Layer: This layer is responsible for getting requests from presentation layer and evaluates the request using data layer. This layer uses system logic layer for computing and evaluating data to be displayed to the user.

System Logic Layer: This layer is responsible for defining system logic. It contains all the components that are related to system logic. It deals with the request from application layer by processing data, making logical decisions, calculations and it is allowed to use network layer to do what is responsible for.

Data Layer: This layer is responsible for management of database; retrieving, updating and storing data. System logic layer uses the data layer and presentation layer uses network layer so data is transferred to the presentation layer and data is presented to the user.

9.7.Layers Context Diagram

Our system stays at the top of the external layers as shown in Figure 32 and has access to them when it needs to use. Wireless porting layer is top external layer that is necessary to be accessed because our system uses a mobile application. At the bottom, kernel stands as external layer. Kernel is used as operating system; our system relies on kernel for core system services such as security, network stack, data management.

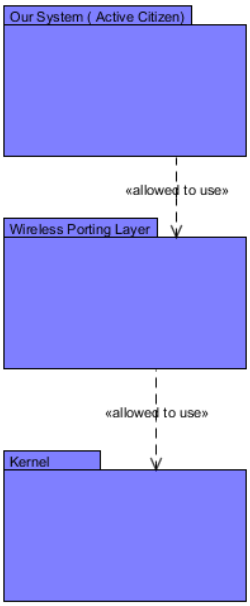

Figure 32: Context Diagram for Layered view of the system

9.8.Aspect View

Active Citizen uses aspects for cross-cutting concerns that are transactional management, exception handling, checking authorization, security, validation and data logging as shown in Figure 33 with what packages they use.

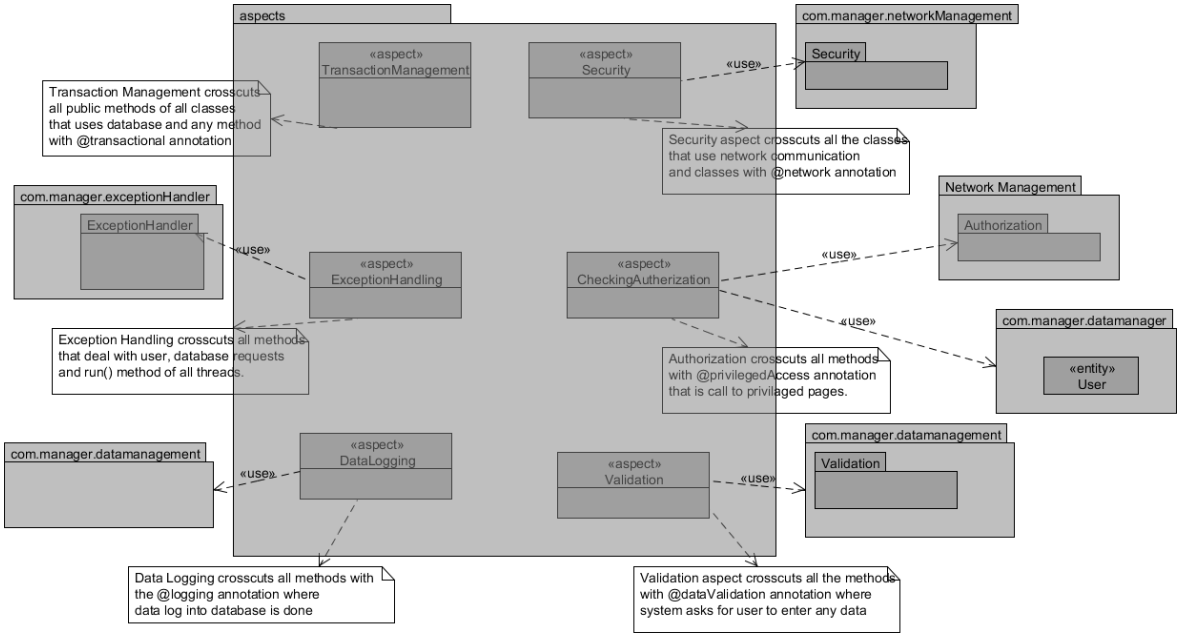

Figure 33: Aspect view of the system

Transaction Management: This aspect is necessary for our system and affects the entire system. It ensures that any operations processed in the system are done successfully or failed as a whole. It provides consistency in the database of the system. If a user of the system makes changes on the data another uses currently changes, it restores the system to the consistent state before the transaction begins. When all requests are done successfully, transaction management ensures that transaction is closed database is released properly.

Transaction Management crosscuts all public methods of all classes that uses database and any method with @transactional annotation as shown in Figure 33.

Exception Handling: This aspect is also applied by all active citizen classes. It throws an exception if undesired action occurs in the system. It makes logs to the database, sends error messages and displays proper message to be shown to the user.

Exception Handling crosscuts all methods that deal with user, database requests and run() method of all threads.

Authorization: The system has to ensure that a user without permission cannot access another users' information and reach private data in the system. Authorization aspect provides this for all classes that have privileged access.

Authorization crosscuts all methods with @privilegedAccess annotation that is call to privilaged pages.

Data Logging: Data logging is another aspect for Active Citizen. Generating log entities is scatting throughout our system i.e. changing one needs updates on other classes related to data logging. Therefore it is crosscutting concern in our system.

Data Logging crosscuts all methods with the @logging annotation where data log into database is done

Security: Security is another concern for our system when data is transmitted via network. Someone can sniff these packages and get information. Therefore security is a common concern for the methods that are responsible for network communication within the system. Security aspect is used for providing secure networking.

Security aspect crosscuts all the classes that use network communication and classes with @network annotation

Validation: Data validation is another aspect for our system. It checks if the data entered by user is valid or not so that valid data is sent to the database and for invalid data, it is asked for user to re-enter the data. Therefore, validation is cross cutting concern that affects the many parts of the system.

Validation aspect crosscuts all the methods with @dataValidation annotation where system asks for user to enter any data.

9.9.Data Model View

In data model view as shown in Figure 34, the entities are illustrated. Local municipality officials and non-governmental organizations are the publics and the public, local municipality officials and admin are the users of the system.

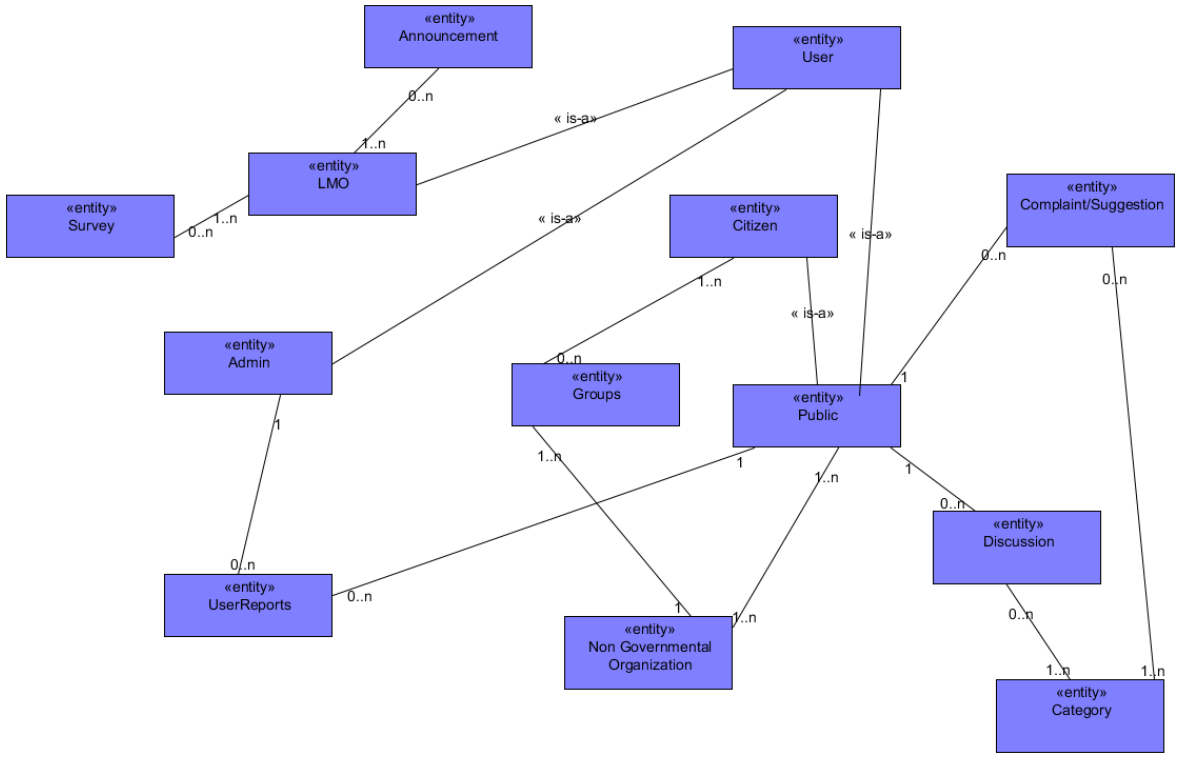

Figure 34: Data model view of the system

Public can have 0 or multiple complaint/ suggestion and discussions whereas complaints/suggestions and discussions can belong to only 1 public. Discussion and complaint/suggestion can have 1 or multiple category. Category can have 0 or many discussion and complaint/suggestion. Local municipal officials have 0 or multiple announcements and surveys that belong to 1 or many officials. Admin can have 0 or many reports and report can belong to 1 user and is reached by 1 admin. Citizen has 0 or many groups and non-governmental organization can announce 1 or many groups.

10. Component and Connector Views

Server-client style, publish subscribe style and multi-tier style are applied to determine run-time behavior and interactions of elements in the system.

10.1. Server-Client View

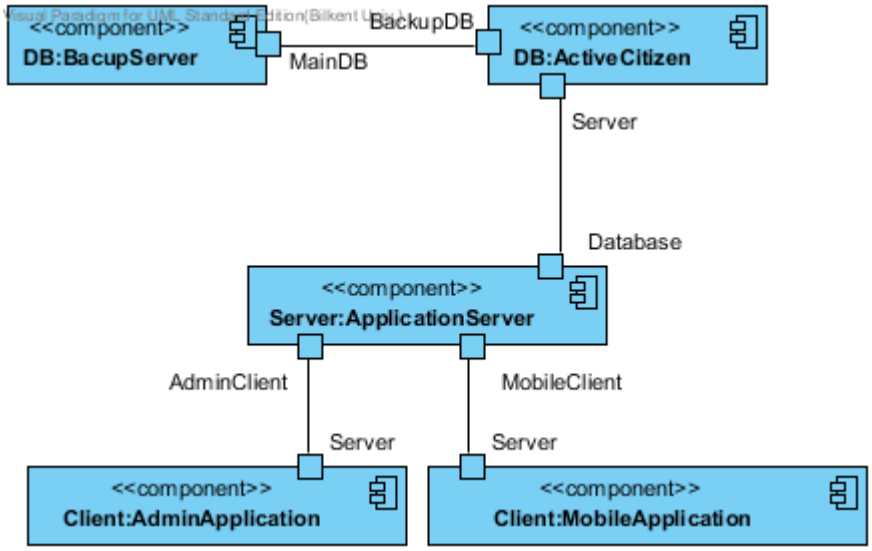

Figure 35: Server-Client View in UML

Figure 35 illustrates Server-Client view of the system in UML as a Component & Connector View. There is one server which provides various services about application logic to client components and another server for database. There is also another database server for backup. There are also many clients in two types: admin application and citizen mobile application which have ports for interaction of application server. Application server has one port for database connection and many ports for admin clients and mobile clients.

In the system, all clients know the identity of application server, but application server does not know the identity of clients in advance. That is, computational flow of the system is asymmetric. As a property of this view, service invocation in the system is synchronous. In other words, the requester of a service from clients waits or is blocked until requested service in application server return a result.

10.2. Publish-Subscribe View

In our system publish-subscribe style is used for announcement system. In Figure 36 publish-subscribe view is shown. Admin Application is publisher, it publish announcements to application server. Application server is both subscriber and publisher; it is subscribed to Admin application and publishes to Mobile Application. Also Application server sends those announcements to database. Backup server for database is also used, main database also store those information on backup server to restore in case of crash.

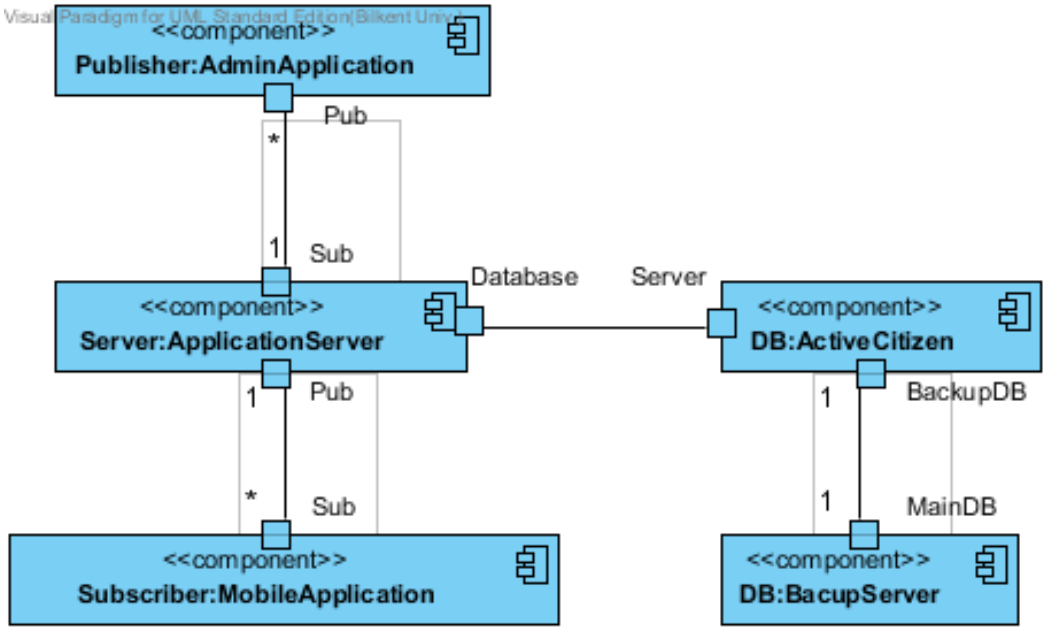

Figure 36: Server-Client View in UML

10.3. Multi-tier View

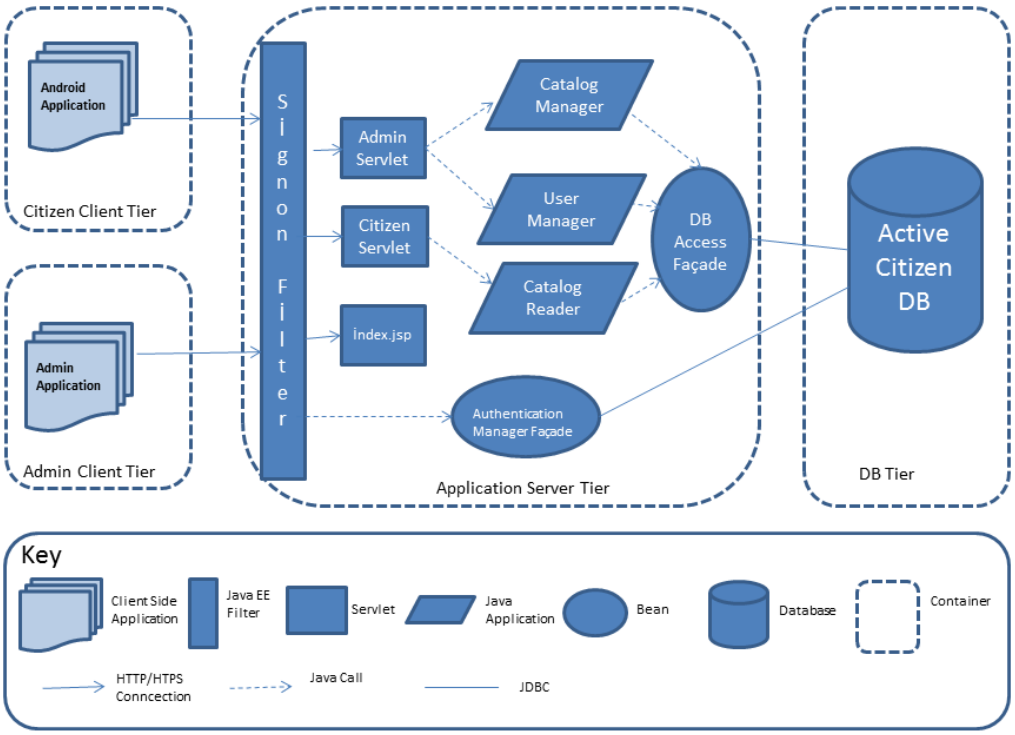

Figure 37: Multi-Tier view of the system

The diagram of Figure 37 reflects multi-tier view of the system using informal notation. It summarizes system partitioning. The diagram shows different platforms where various parts of the system run on. For instance, citizen application executes on Citizen Client Tier which is mobile phone with Android OS and admin application executes on Admin Client Tier, which is a PC. Logical group of components about application services runs on application server tier. Connectors described in 'Key' exist only between components in the same tier or adjacent tiers as observed in diagram.

11. Allocation Views

Allocation views are applied to mapping software architectures onto its environment from three aspects: hardware elements as deployment view, file management as install view and organization team as work assignment view.

11.1. Deployment View

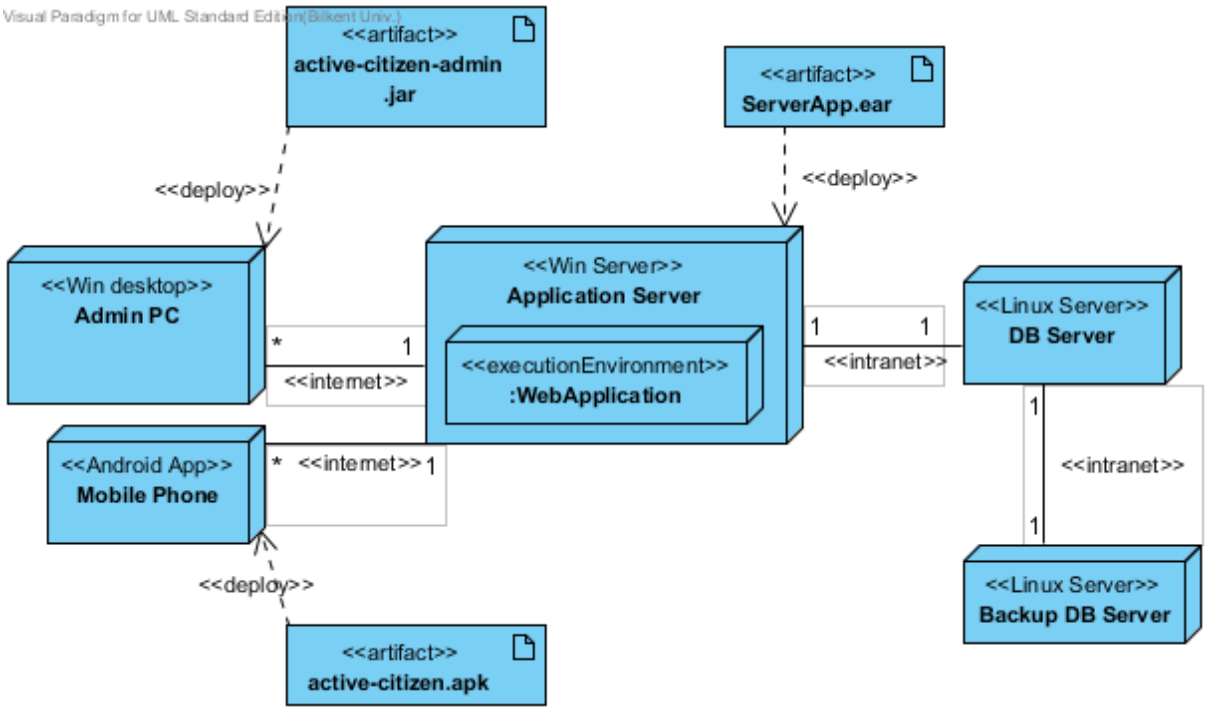

Figure 38: Deployment view of the system

Figure 38 shows the hardware of the system and the software installed on that hardware in UML. There are four five of hardware components which Mobile Phone, Admin PC, Application Server and Database Server and Backup Database Server. Active-citizen-admin.jar, which is a file typically used to aggregate many Java class files and associated metadata and resources, installs on PC with Windows. Active-citizen.apk, which is Android application package, deploys to mobile phone with Android OS. ServerApp.ear, which is a file for Java EE packaging, deploys to Application Server.

11.2. Context diagram for deployment view

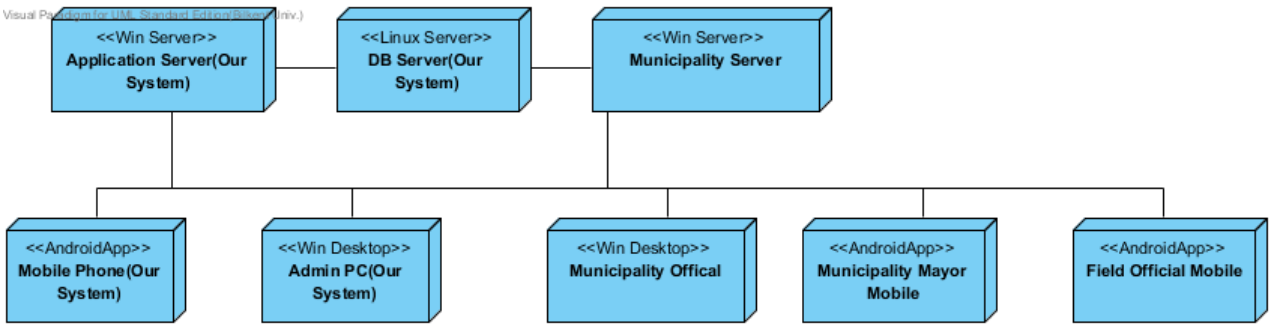

Figure 39: Context Diagram for deployment view

The diagram in Figure 39 illustrates context diagram of the system for deployment view. Hardware's of our system are mobile phone for android application, windows PC for admin

application, windows server for application logic and Linux server for database. The database server in our system is also allocated to external hardware's which are windows server for current municipality operations, windows PC for municipality officials and mobile phones with Android OS for field officials.

11.3. Install View

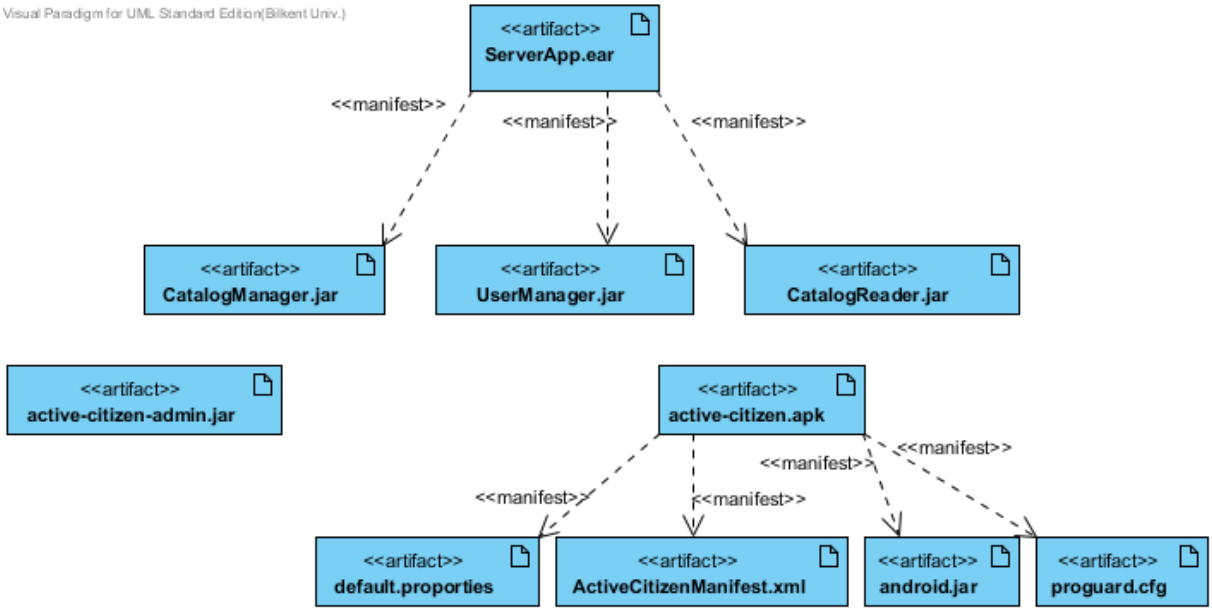

Figure 40: Install view of the system

The view in Figure 40 shows how the installed system is organized as a structure of files and folders. Catalog Manager, User Manager and Catalog Reader java archive files are manifested in the package ServerApp.ear which runs on application server. Active-citizen.apk, which executes on mobile phone, manifests android.jar as Android OS properties, an xml file as graphical user interfaces and proguard.cfg as required library.

11.4. Work Assignment View

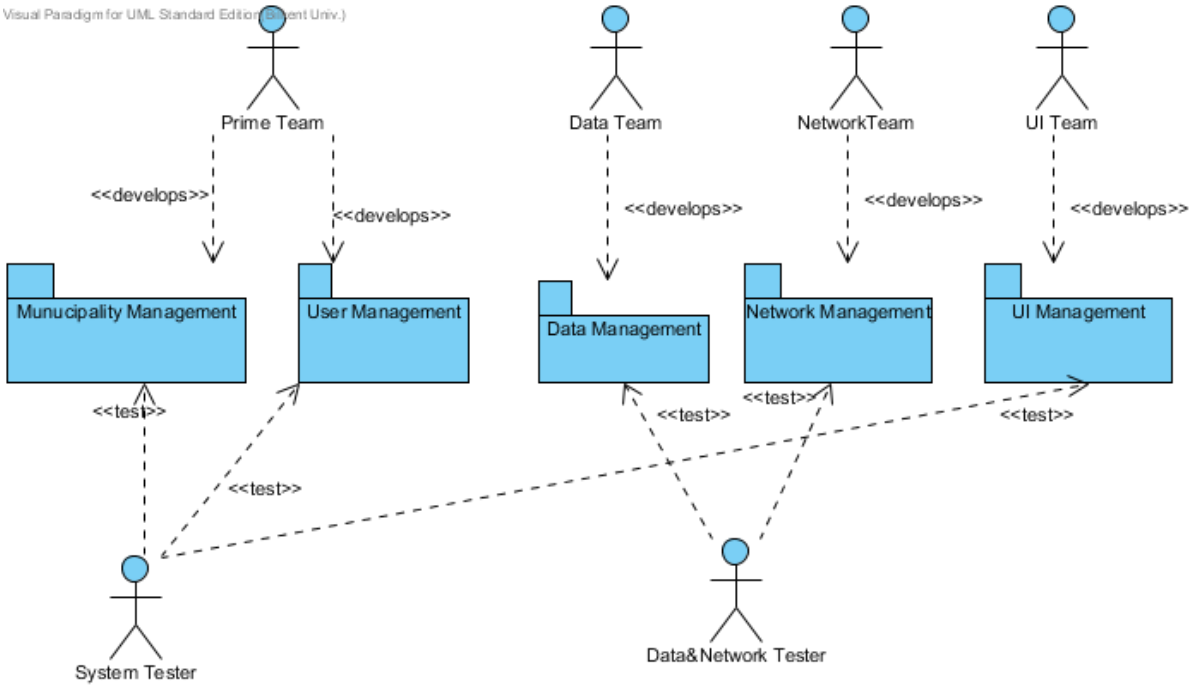

Figure 41: Work Assignment View of the system

Work Assignment view as shown on Figure 41, illustrates the mapping of subsystems onto the people, groups, or teams tasked with the development and testing of those subsystems in UML.

12. Evaluation of software architecture

12.1. Scenario-based Architectural Analysis (SAAM)

Step 1. Describe candidate architectures

We have already described the candidate architectures as views and beyond approach in our system in the sections above. We have applied decomposition, uses, generalization, layers, aspect and data model view as module views, client-server, publish-subscribe and multi-tier as C&C views and deployment, install and work assignment views as allocation view. We have chosen to apply these architectures based on the functional and quality requirements.

Step 2. Develop and prioritize scenarios

In Table 15, descriptions of scenarios with their priorities are given and stakeholder match is shown. The important stakeholders for the system are:

- End users
- Database Administrator
- Software Architect
- Software Designer
- System Maintainer

- Implementer
- System Integrator
- Project Manager
- System Engineer
- Tester

The roles of stakeholders are described in requirement analysis stage of the report. In the scenario based architectural analysis, their roles' match to scenarios is shown.

Scenario with priority	Description	Stakeholders
1	Recover the system in case of server crash	Maintainer
2	Store password in the database in a secure manner	Database Administrator, Maintainer
3	Serve multiple citizens at the same time	Maintainer, System Engineer, Database Administrator
4	Store images in the database consistently with suggestions /complaints	Database Administrator, Maintainer, End User
5	Resistance to upgrade in Android OS release	Maintainer
6	Port system to different devices with Android OS	Maintainer
7	Port system to different operating systems	Maintainer
8	Respond to citizen requests 7/24 hours	System Engineer, Designer
9	Remove user from the system	Database Administrator, Maintainer
10	Isolate different user groups	End User
11	Change screen resolution	Maintainer
12	Change color of widgets on the background	Maintainer
13	Official views the current ratings of all municipalities	End User
14	Add voice of control to the system	Maintainer, Integrator

Table 15: Scenario definitions with priorities and stakeholders

Step 3. Perform scenario evaluation

Scenario-based Architectural Analysis method is used to perform architecture evaluation based on the scenarios derived by the stakeholders of the system. All architectural scenarios based on quality and functional requirements are gathered, their interactions are revealed and overall evaluations are generated. In Table 16, the scenario descriptions and the evaluation results are shown. Direct scenarios can be executed by the system without any modification, whereas in order for

indirect scenarios to be performed by the system, the changes described in the last column of the Table 16 should be applied to the architecture.

Scenario	Description	Direct/Indirect	Changes Required
1	Serve multiple citizens at the same time	direct	
2	Change color of widgets on the background	indirect	This requires change of UI Management packages in a way that change request is taken from the user and applied to the background image. This can be done by adding a component "Background Manager" to the UI Management module.
3	Port system to different operating systems	indirect	The platform-independent modeling layer between the application layer and presentation layer must be added. The changes will be made to Process Manager in a way that the new package which fulfills the platform mapping functionality will be added in order to provide the Model-Driven Architecture approach.
4	Port system to different devices with Android OS	direct	
5	Resistance to upgrade in Android OS release	direct	
6	Remove user from the system	direct	
7	Store images in the database consistently with suggestions /complaints	direct	
8	Isolate different user groups	direct	
9	Store password in the database in a secure manner	direct	

10	Recover the system in case of server crash	direct	
11	Official views the current ratings of all municipalities	indirect	Each municipality uses its own application server and in order to access the data for other municipalities, Data Management package in manager.ejb must be adjusted in a way that all municipalities are connected to this main server. The capacity of the module may be extended or new Data Management component can be added to work synchronously with the already existing one.
12	Respond to citizen requests 7/24 hours	direct	
13	Add voice of control to the system	indirect	UI Management package in manager.ejb should be changed in a way that "Voice Recognition Module" must be added to the UI management package. If the user chooses the voice recognition option, the "Voice Recognition Module" will be activated.
14	Change screen resolution	indirect	When the picture is minimized and then maximized, the image resolution should not be worse. The operation should be handles in a way that in the UI Management package, "Image Resolution Manager" package should be added.

Table 16: Scenario-based Architectural Analysis

Step 4. Reveal scenario interactions

Each indirect scenario defined in Table 16 requires a change in some architectural components. The changes will be made to UI Management, Process Manager and Data Management components in the existing system based on the evaluation results in Step 3. Addition of new components to UI Management, Process Manager and Data Management

modules yield the scenario interaction table in Table 17. In the table, the number of changes required to each module is shown and the results are derived considering the “uses” relation among modules.

Module	Number of Changes
UI Management	3
ProcessManager	1
Data Management	1

Table 17: Scenario Interaction Table

Step 5. Generate overall evaluations

The attributes that are related to the user interface design require additions to UI Management component. In addition to that, porting system to different operating systems problem which is related to the portability attribute and for officials to view the current ratings of all municipalities problem which is related to scalability attribute require additions to Process Manager and Data Management components subsequently. The reason behind this is that the existing architecture does not execute those scenarios. In order to make the system to serve Scenario 2, 13 and 14 in Table 16, we have to add some modules to UI Management module. This module has the largest scenario interaction and requires most number of changes. Therefore UI Management module should be decomposed into 3 sub modules; Background Manager, Voice Recognition Module and Image Resolution Manager.

In order to make the system to serve Scenario 3 in Table 16, we have to make changes to Process Manager Module. In addition to Network, Municipality, Data and UI Management modules, the new module which fulfills the platform mapping functionality should be added in order to provide the Model-Driven Architecture approach to Process Manager Module.

In order to make the system to serve Scenario 11 in Table 16, we have to make changes to Data Management module. The capacity of the module may be extended in a way that all municipality ratings are managed in that module or new Data Management component can be added to work synchronously with the already existing one.

12.2. Utility Table (ATAM)

Architectural trade-off analysis method is applied to the architecture of the system in order to evaluate consequences of architectural decisions from the aspect of multiple quality attribute requirements. This method not only shows to what extent the architecture satisfies particular quality goals, but also it provides us to understand the interaction between those quality goals, i.e. the tradeoff between quality attributes. Then, quality attribute utility table is generated as shown in Table 18. The defined quality attributes are refined according to the system and the attribute refinements are evaluated according to the architecture. The derived scenarios are prioritized along two dimensions, importance and the degree of difficulty. For example for the performance quality attribute, for the response time, the derived scenario has high (H) importance for the success of the system and medium (M) degree of difficulty in terms of implementation.

Quality Attribute	Attribute Refinement	Scenarios
Performance	Response Time	Server gives response to clients in less than 50 ms.(H,M)
	Throughput	Server allows 100 transactions in a second. (M,M)
Sustainability	Resistance for new releases of Android OS	When user of mobile phone updates its OS, application can be still runnable. (H,L)
	Low cost for sustaining system	The cost of maintaining servers and DB refinement should not be extremely high. (M,M)
Availability	Server uptime	The system should give response 7/24. (H,L)
Modifiability	Changeable User Interfaces	User interface could be different for different needs of municipalities.(H,M)
	Adaptation of modules	Modules should be adapted according to different municipalities.(H,M)
Security	Storing passwords	Passwords should be stored in encrypted and salted way.(H,L)
	Access Control	Authorization should be provided in a way that each type of user should access the data that she/he is allowed to reach. (H,M)
Adaptability	Different devices support for mobile application	Mobile application should execute on different mobile phones. (M,H)
	Different OS support for PC admin application	Admin application is able to run on PCs with different OS such as Windows, Linux, Mac and Solaris. (M,H)
Scalability	Usability for all municipalities	The system should be scaled for all municipalities. (M, M)
Consistency	Image-indexing	Images should be stored in the database in a way that they are displayed under the topic where they are uploaded. (H,L)
	Comments/complaints indexing	Comments/ complaints should be stored in the database in a way that they are displayed under the topic where they are written. (H,L)
Recoverability	DB Recoverability	In case of crashing DB, database should be restored from the back-up. (H,L)
	Server Recoverability	In case of crashing application server, there should be a back-up server to give response. (H,L)
Traceability	Tracking Work-flow	User can trace status of his/her inputs like comments/complaints to the system. (H,M)

Table 18: Utility table

13.Conclusion

In this report, we focused on the software architecture design of our system which will allow the citizens to follow the civic issues and be able to communicate with the governing bodies via an online mobile interface by removing the time and place obstacle. After we explained the context and goal of the system, we specified the steps followed in designing the software architecture, which are determining context diagram, describing basic problems and the specified domain. Then, based on these steps, we explained requirement analysis, technical problem analysis and domain analysis respectively. After all, thereby benefiting from these works, we prepared conceptual architecture design of the system.

While we were preparing this report, we have practically learnt how to reach software architecture design from requirements. Adapting from domain-driven approach, we have practiced how to make technical problem analysis, domain analysis and conceptual architecture design on our system. In technical problem analysis we observed that dividing main problem into sub-problems and prioritizing them on our case obviously eases to shape the software architecture design of the system. After that, we have learnt how to identify domain and evaluate knowledge sources by mapping domains with technical problems. Based on solution domain concepts as reflection of solution domains on our system, we have specifically learnt how to prepare sub-level conceptual architectures for solution domains.

Additionally, we have practiced view and beyond approach applying each view to our system. We have experienced that module views document the principal units of implementation, C&C view shows the elements that have run time behavior in the system and the last view; allocation view shows how the system is mapped onto its environment. With each component of views, the software architecture of the system is shaped. We have also extracted the context diagram for specific views, showing external entities of the system with different element and relational type for each view. Finally, we have experienced that evaluating architecture is necessary to ensure the quality of software architecture. We have tried to analyze for quality of the system, potential risks and how the system will evolve by software architecture evaluation. For this purpose, Scenario-Based Architecture Analysis Method and Architecture Trade-off Analysis Method have been used so scenarios help to understand the system context and detect errors in the architecture.

Moreover, we got used to benefitting from tool support to draw diagrams since we used Visual Paradigm and UMLet tool to present software architecture design with stereotyped classes, packages, use cases and dynamic models such as state and sequence diagrams based on semiformal notation, UML for all diagrams but for multi-tier view, we have used informal notation to represent.

In this report, we have faced various challenges. We had a serious difficulty in directly adapting synthesis based software architecture design approach for our case. There was no alternative to apply alternative space analysis. Hence, we did not apply alternative space analysis phase for our project but other phases of this approach such as technical problem analysis and domain analysis were directly applied to make software architecture design. It is

also challenging to prepare sub-level conceptual architectures for the solution domains of our system since our system is not complex to divide many sub-levels. Additionally, we had some difficulties on finding indirect scenarios for the same reason that our system is not that complex to require changes on the packages and system is considered to design for adaptability and portability so it can be integrated to new environment without much change.

14.References

1. Paulo Merso, “Representing Aspects in the Software Architecture – Practical Considerations” http://www.cse.cuhk.edu.hk/~elisa/EA/Merson05_EAworkshop.pdf
2. Ali Mesbah, “Crosscutting Concerns in J2EE Applications” http://homepages.cwi.nl/~arie/papers/aopetstore/wse05.pdf
3. Microsoft Corporation, “Three-Layered Services Application” http://msdn.microsoft.com/en-us/library/ff648105.aspx