Personal Paper for

Software Engineering I

WS 01/02

Roman Gross

Contents

Chapter 1

Introduction

For the subject Software Engineering we have to develop a program named ImmoWarrior. In the next paragraph I want to list the main task given to our group from the virtual company e-Immo.

The program should support the estate agency in the process of collecting information and representing this information. The process has to be kept as easy as possible. The potential user of the system should be the collector and the recaller of the information, whereas the recallers can be persons or foreign systems.

Two possible scenarios can be the result of this:

- Collected information is passed on other systems by ImmoWarrior (using XML)
- Collected information is passed on single persons. Therefor, an appropriate preparation of the information is needed

Furthermore, we should keep an eye on the maintainability of ImmoWarrior (e.g. no installation of system components).
Another point is that ImmoWarrior should provide the possibility of manipulating the preparation of information for single user. This process has to be kept easy, too.

Chapter 2

CRC Session

2.1 Preparing the CRC Session

For the first lesson we have to prepare a CRC session. CRC is the abbreviation for Class, Responsibility and Collaborator. The requirements for such a session are crc cards, which have to look like the example below.

Classname	
Responsibilities	Collaborators

What is a:

Class: A class is a collection of similar objects. Potential classes are actors (people or organizations interacting with the system), events, concepts or major user interfaces. It is very important to find three to five main classes to describe the core of the system.

Responsibility: The things a class knows and does. It describes what the class is all about.

Collaborators: If a class is not able to fulfill its responsibilities, it has to collaborate with other classes.

The main purpose of a CRC session is to define the user requirements for the system. A collection of CRC cards is called a CRC model. Such a CRC model describes a part of the problem domain or covers the whole problem domain.

A CRC modeling team consists of the following persons:

Business Domain Experts (BDE's): The BDE's are the future user of the system. They have experience in the problem domain. Good BDE's know their business, think logically, can communicate well and they are willing to invest the time in software developing.

Facilitator: First of all, a facilitator should have good meeting skills and has to understand the entire software process, because he runs the meetings. He or she asks valid and intelligent questions to help the BDE's explore the various aspects of the system. Furthermore, a facilitator has to make sure that the CRC cards are filled out properly.

Scribes: In every meeting there are one or two persons writing down the business logic and anything else not written down on the CRC cards. Scribes often need to ask questions to determine what the BDE's mean. So they have good communication skills.

Observers: They do not actively participate in the meetings. Those persons sit in the back of the room and observe the meeting (e.g. for educational purpose).

Our first job for the meeting with the representative of our client on Wednesday is to plan a CRC session. Below I have written down some suggestions for CRC cards describing the problem so far.

Classname: **Real Estate**	
Responsibilities	**Collaborators**
size location is stored	information storage

Classname: **User**	
Responsibilities	**Collaborators**
name password log in enter real estate view real estate	user interface real estate

Classname: **User Interface**	
Responsibilities	**Collaborators**
style position display information react on user input get data from info storage	information storage user

Classname: **Information Storage**	
Responsibilities	**Collaborators**
real estate data user data	real estate user

2.2 The Proseminar on October, 31th, 2001)

In this meeting we have analyzed and refined the problem domain and figured out the basic classes. We asked questions and updated our CRC cards during this process.

The results of this meeting:

1. Relevant information should be obtained by the user via Internet or cell phone.
2. The information should be sent to the customer either automatically by the system (push) or on individual request (pull).
3. Clients as banks, authorities and private persons get only one account.
4. The system should provide an access-control-system.
5. Private persons receive their username/password by e-mail.
6. Private persons have to pay provision for publishing their information
7. There should be the possibility to include different multimedia-elements (pictures, movies, sounds,...).

Some possible classes we worked out: Session, User Interface, Account, Permission, Information Storage, Real Estate, Editor, Private Person, Prospective Buyer

2.3 The latest CRC model

In this section you see the latest view of the CRC model. They were updated and refined during the conceptional and the analysis phase of the development process.

Classname: **Admin**	
Responsibilities	Collaborators
access user interface view free offers log in configurate account create/modify account search for real estate view real estate info place/edit/delete real estate log off	user interface

Classname: **EditorAndUser**	
Responsibilities	Collaborators
access user interface view free offers log in configurate account search for real estate view real estate info place/edit/delete real estate log off	user interface

Classname: **Editor**	
Responsibilities	Collaborators
access user interface view free offers log in configurate account place/edit/delete real estate log off	user interface

Classname: **User**	
Responsibilities	Collaborators
access user interface view free offers log in configurate account search for real estate view real estate info log off	user interface

Classname: **Guest**	
Responsibilities	Collaborators
access user interface view free offers request new read account	user interface

Classname: **User Interface**	
Responsibilities	Collaborators
read input forward data show info start session	guest user editor editoranduser admin session

Classname: **Session**	
Responsibilities	Collaborators
create/modify/delete account initiate sending of account info forwards data to user interface initiate search send real estate info close session access info storage request account info	user interface account send account info search for info send real estate info info storage

Classname: **Account**	
Responsibilities	Collaborators
access info storage modify account info forward account info	session info storage

Classname: **Information Storage**	
Responsibilities	Collaborators
store account info store real estate info forward data	session account search for info

Classname: **Search For Info**	
Responsibilities	Collaborators
perform search forward search results	info storage session

Classname: **Send Account Info**	
Responsibilities	Collaborators
send username to e-mail/cell phone send password to e-mail/cell phone send account privileges to e-mail/cell phone	account session

Classname: **Send Real Estate Info**	
Responsibilities	Collaborators
send info to e-mail/cell phone look up account info	session account

Chapter 3

Defining Use Cases

The essential use case model gives an technological independent view of how the further user interacts with the system. Use case diagrams are a part of the use case model. Such a diagram consists of actors, their associations, a system boundary box, and packages. A use case is a sequence of actions, drawn in a horizontal ellipse and an actor is something (a person, foreign system...) interacting with the system. This association is modeled by an arrow, in which an association can also appear between use cases.

3.1 Acting out Use Case scenarios

Use case scenarios are acted out by the facilitator and the BDE's. At the beginning the facilitator hands out the CRC cards to the BDE's. When acting out use case scenarios, you can stick to a sequence of six steps :

1. **Call out a new scenario:** This is done by the facilitator. He reads a scenario description to the group. After this, the involved classes have to be found.

2. **Determine which card should handle the responsibility.**

3. **Update the cards whenever necessary.** If a new responsibility comes up, the right class has to be updated or a new class with this responsibility has to be created.

4. **Describe the processing logic.** When the ball is thrown to someone, he or she should describe the business logic for the responsibility step by step. Here, the business logic is described and should be written down by the scribes.

5. **Collaborate if necessary.** A class often cannot fulfill all the responsibilities, so it has to collaborate with another one. The ball is thrown to this class

6. **Pass the ball back when done.** If the description of responsibilities of one class is done, the ball is passed back to the class collaborating with the actual one. If the ball goes back to the facilitator, the use case is finished.

3.2 Use Case Descriptions

Name: View Free Offers

Basic Course of action:

1. A person visits the home page.
2. The guest can view some free offers.

Name: Log in

Description: The registered user enters username and password

Preconditions: A valid account

Postconditions: the user is logged in and can take actions according to his privileges

Basic Course of action:

1. The user types in the user name and the password.
2. The system verifies the account.
3. If he is eligible to log in the system displays the main menu.

Name: Create Account

Description: A person wants to have access to all real estates

Preconditions: None

Postconditions: A new user will be created and the data is sent to his e-mail address or his mobile phone

Basic Course of action:

1. The person has to enter the main user interface (e.g. home page).
2. He has to click on the *Become a Member*-button.
3. Now the person has to fill out a form.
4. If he submits, a new user is created and its data stored on the information storage
5. The user data (username, password) is sent to the user (e-mail, sms)

Name: Create Real Estate

Description: A user places a new offer in the system

Preconditions: User with the privilege to place a new offer

Postconditions: A new real estate is offered by the system

Basic Course of action:

1. The user has to log in.
2. The user opens a form where he can place pictures, a description of the real estate and some multimedia files.
3. After submitting a new real estate offer is stored on the information storage.
4. The data is sent to several user.

Name: Search For Real Estate

Description: Searching for real estates with predefined search criteria.

Preconditions: Valid read account

Postconditions:

Basic Course of action:

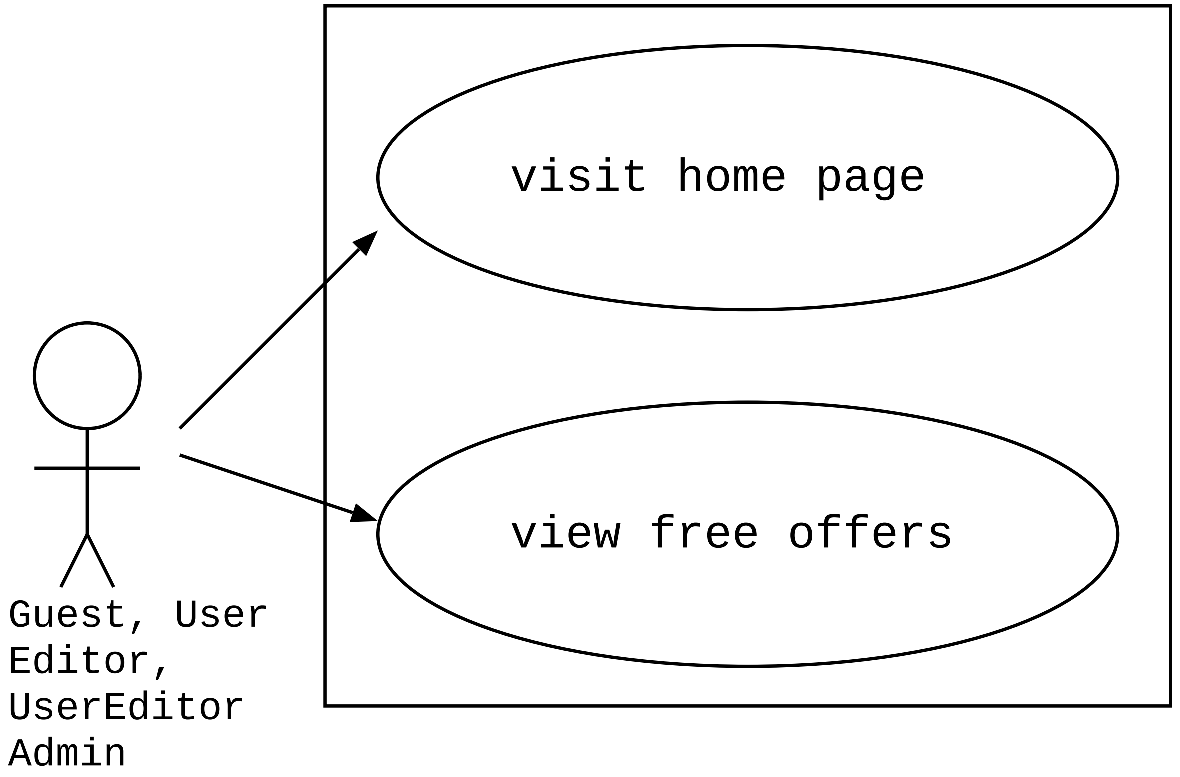
1. The user logs in
2. He has to click on the *search for real estate*-button in the main menu.
3. A form with search criteria is displayed where the user can enter one or more keywords.
4. If the system finds suitable real estates, a list of them is displayed, else an *No Match* information is displayed.

In a group discussion we determined five different actors which mainly differ in the permission they have in the system. The enumeration below describe them from the most general, the guest, to the admin.

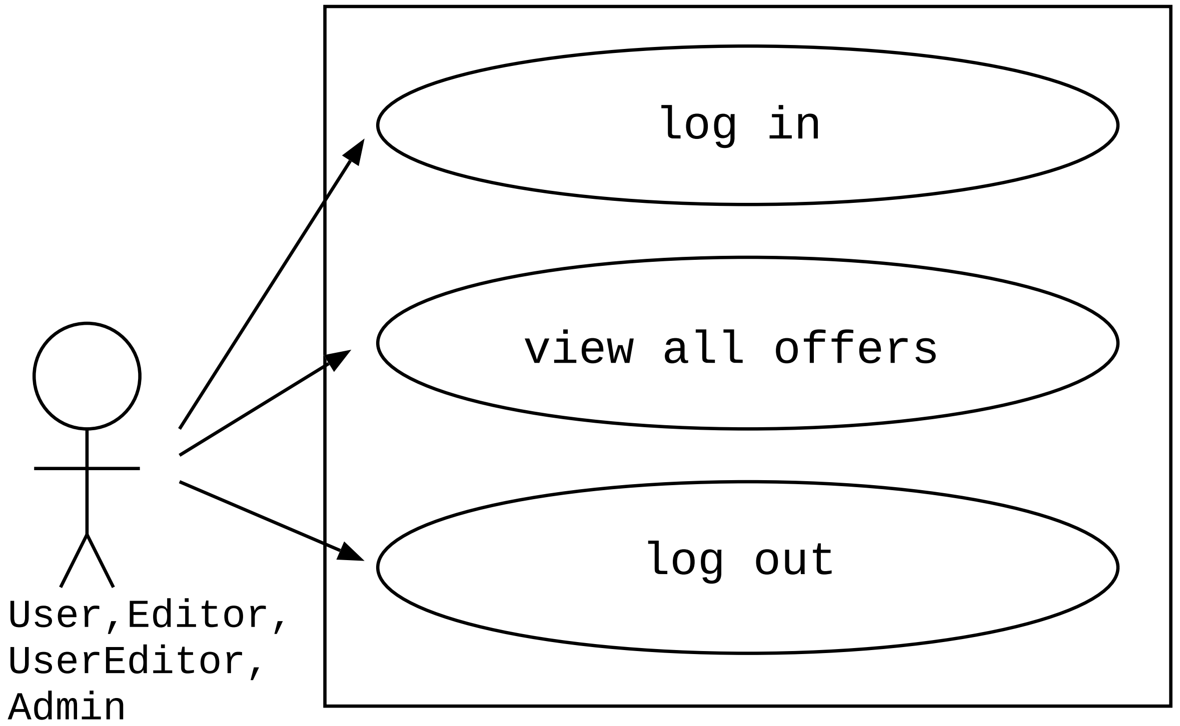
- **Guest:** The Guest is the most general actor dealing with the system. So he is just allowed to view the free offers and he can apply for a new account.
- **User:** A User already has a read account, so he can log in and view all real estates offered by the system.
- **Editor:** Like the User, an Editor is able to log in, but he is only allowed to place new real estates and not to view them, except his own.
- **UserEditor:** The UserEditor is a fusion of the User and the Editor. He is allowed to view all the offers and to place new offers.
- **Admin:** An Admin has all privileges the other actors have and additionally has some special rights as far as account management is concerned.

3.3 Use Case Diagrams

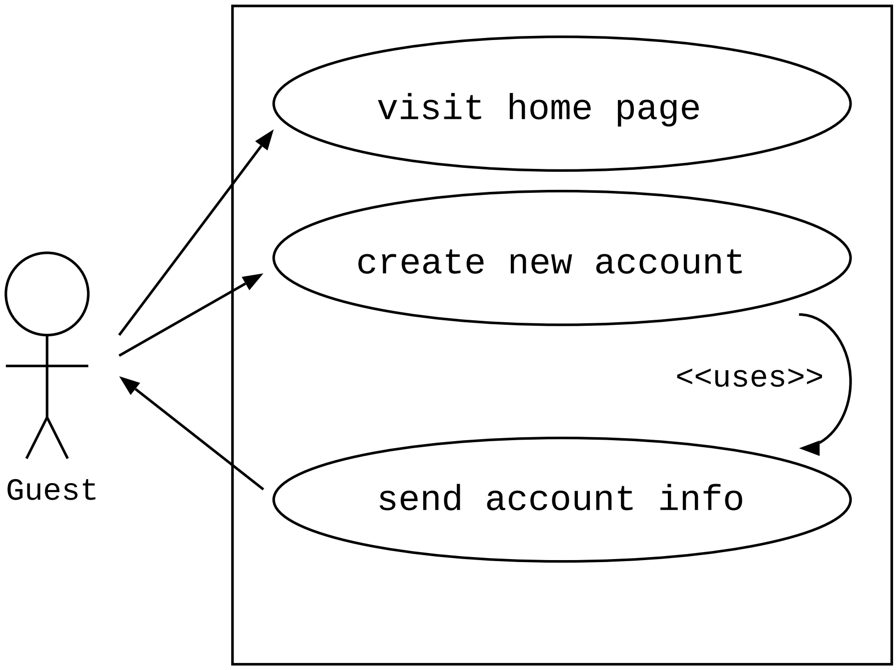
View Free Offers



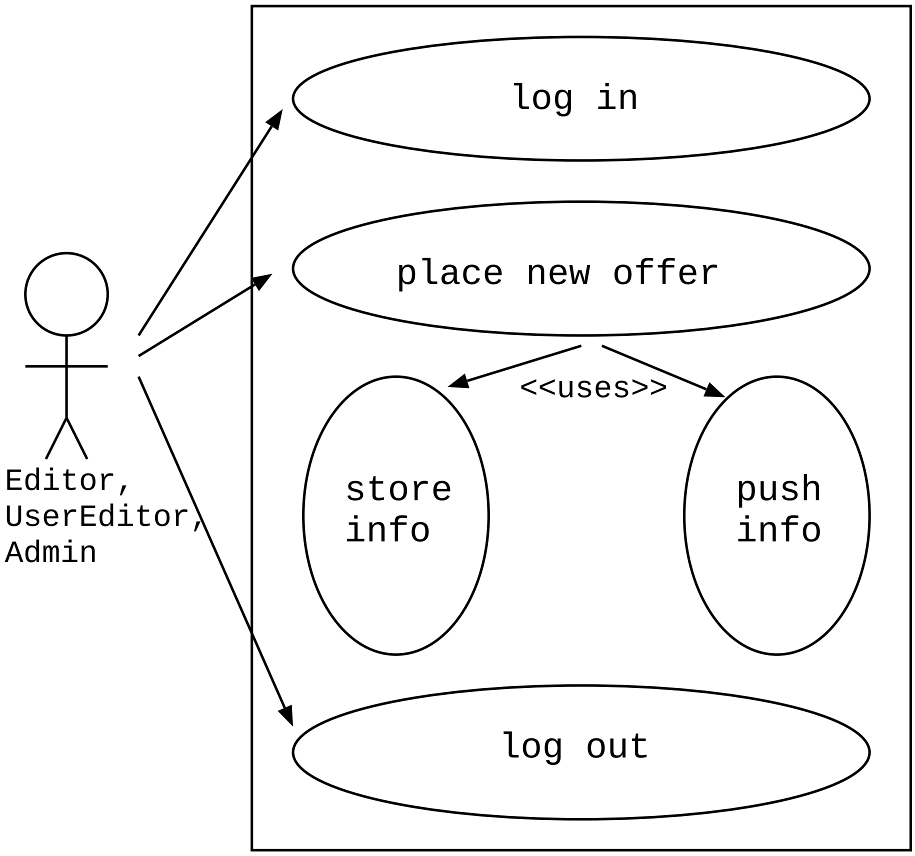
Login/View All Offers



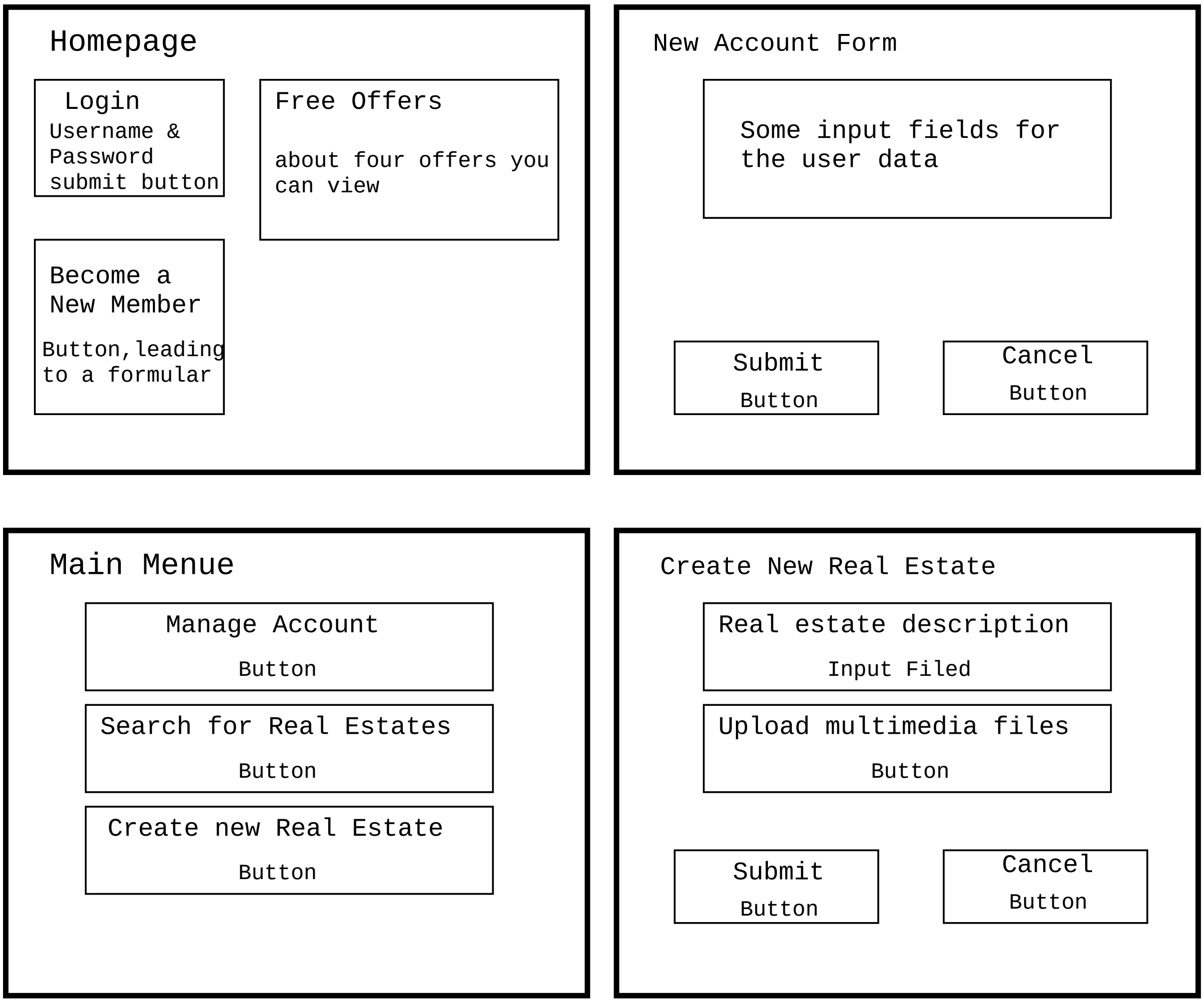
Create New Account

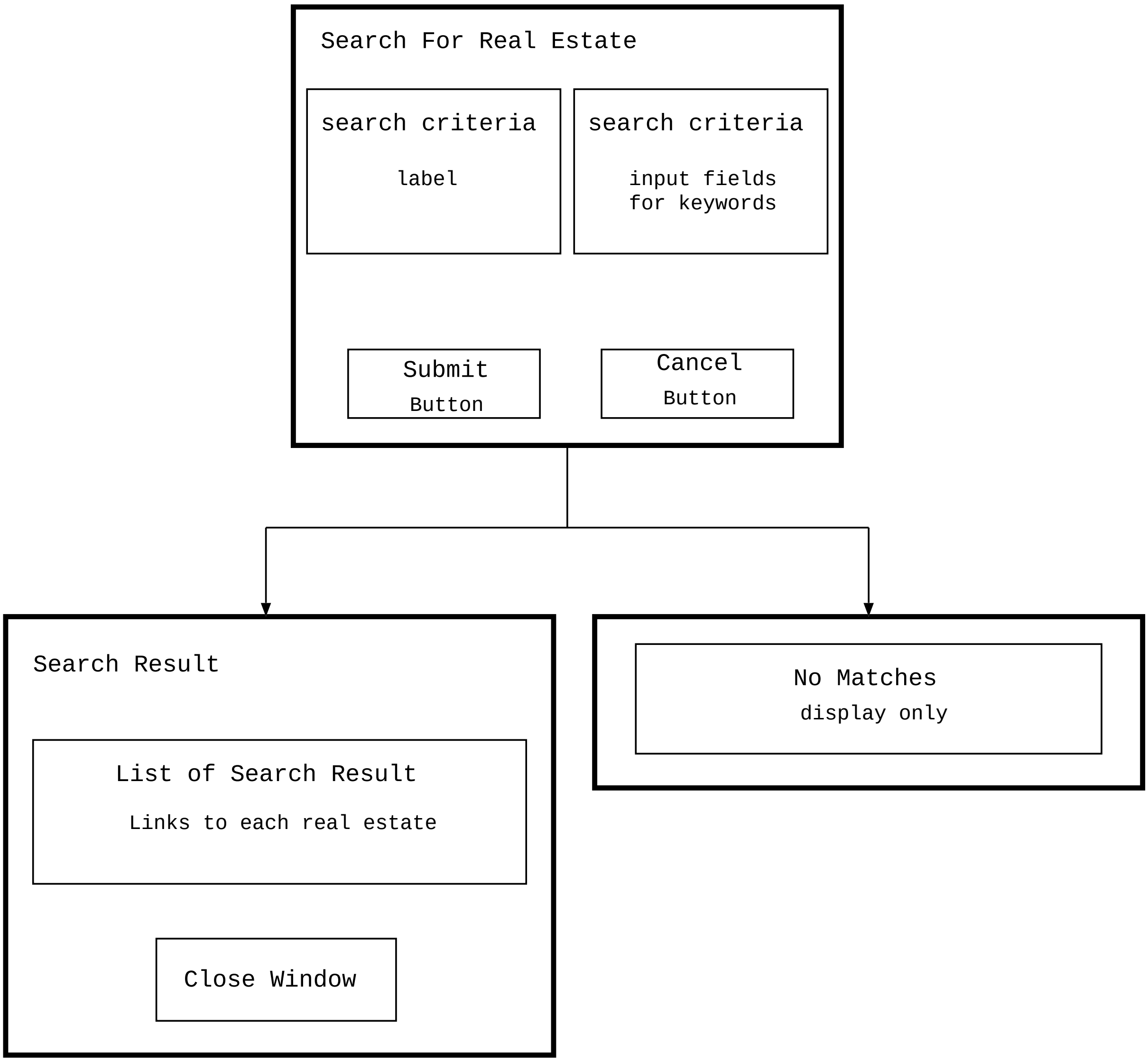


Place New Real Estate



3.4 Essential User Interface





3.5 The Proseminar on November, 07th 2001

In this lesson we discussed the refined problem domain and talked about some use cases we found out and created new one (offer placing, create account). As another result we found out new classes (account information, user).

3.6 The Proseminar on November, 14th, 2001

Our group leader had to present the essential use cases and the suggestions of our basic user interface. We agreed that the actors can be designed hierarchical using inheritance, starting with the most general, the guest, which only can view the free offers and create a new read account.

Chapter 4

Class Diagrams

4.1 What are Class Diagrams

The first two chapters were about gathering information and the conceptual phase of the software development process. Class diagrams are a part of analysis phase. The CRC cards, the essential user interface and the essential use cases are the basis for this.
Class models show the classes of a system, their relations, methods and attributes. They are used to describe the conceptual model during analysis phase. The relations can be one to one, one to many, many to one, many to many, where many can also mean zero. A relation between classes can be an inheritance, where a subclass inherits all methods and attributes from a superclass and can additionally have own methods and attributes. Another relationship is an association.
Every CRC card is modeled as a class in the class diagram except the cards representing actors, because they only exist in the real world.

The classes we figured out and their functions are described as follows:

4.2 The Class diagram

userInterface: reacts on events from outside the system like clicking on a link or button in the browser. The user interface collaborates with the session to get data from the dataBase (e.g. account) which it has to display. This data is forwarded to the layout.

layout: Objects of this class contain information like the style of a window and how the information has to be displayed.

session: The session is a central point of our system. It collaborates with userInterface, accountController and realEstateController and uses their functions to fulfill its responsibilities. Main methods are login, saveAccount, saveRealEstate, createAccount, getAccountInfo, getSearchresult.

accountController: This class manages all the accounts stored on the information storage. Therefor it has to collaborate with the databaseInterface. Furthermore it forwards account information to the session (e.g. when a user logs in) and also gets information from the session (store a new or changed account).

accountInfo: An instance of this class contains the information of one account. In the proseminar we created three new classes, associated with accountInfo through one to one relations. Those new classes are address (of the user), connection with the attributes phone, e-mail and cell phone, and privileges.

realEstateController: This class has the same meaning for real estates as accountController has for accounts. It provides functions for creating, modifying, deleting real estates and with its help the session can search for real estates.

realEstateInfo: Contains all information of a real estate like the description, links to multimedia files...

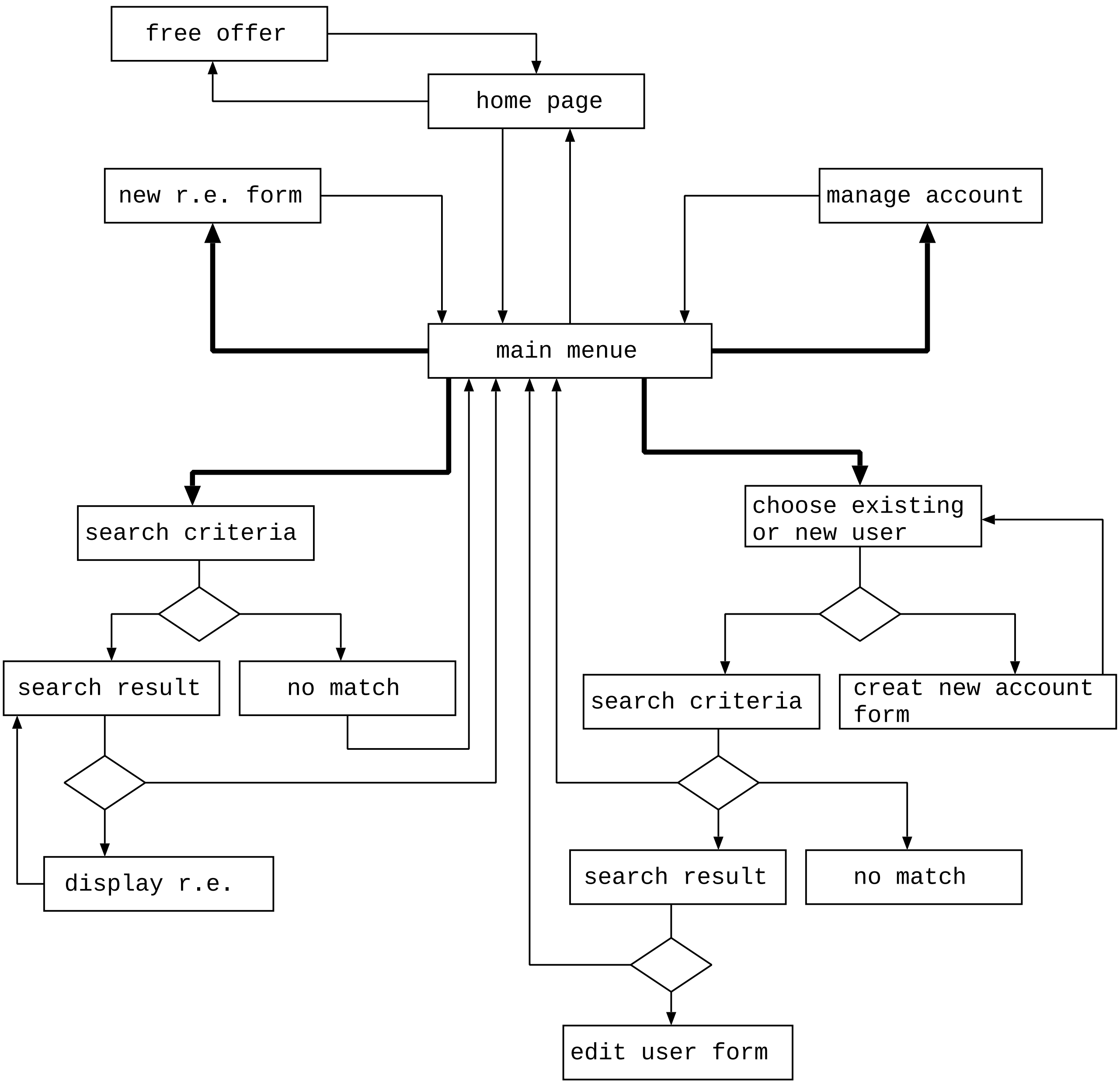
dataBaseInterface: The database manages the access to all the user and real estate data being on the information storage. A possibility to handle this are SQL statements.

4.3 The Proseminar on November, 21th, 2001

Our class diagram was presented. An improvement coming up during the session was to drop all the child classes from userInterface and replace them by a class called layout, containing all the information and methods to display a user interface. As I already mentioned above, the attributes address and privileges of account should be modeled as own classes.

Chapter 5

User Interface Prototyping



The figure above gives an overview of the possible paths you can go through our program. The entry point in the html version is the home page. there you can see the free offers, create a new account or, if you already have an account, log in.

In the main menu you can choose out of five possibilities according to your privileges:

- create new real estate
- manage your account
- search for real estate
- administration
- log out

For further information i want to refer to the following links:

- HTML version: http://www.cosy.sbg.ac.at/ ~ mpober/se1/se_html/
- WML version: http://immowarrior.wap3.de/

5.1 The Proseminar on December, 05th, 2001

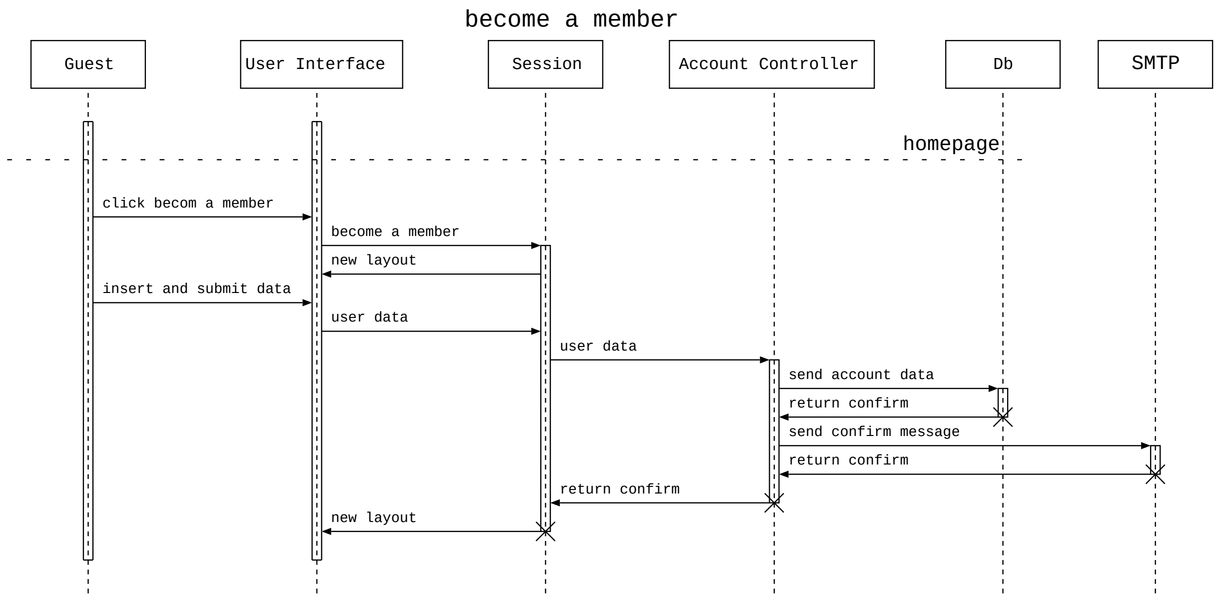
The group leader presented our html and our wml user interface. We found out that we have to make some improvements. Our home page did not like resolutions below 1024*768 and refused to run on several browsers. As a solution to this, Java Script were suggested. Furthermore, we found out that it would be better to display the input search criteria on the mobile phone when the user decides to type in more than one. Another very important thing we have to improve is that a user with just wright permission should be able to view his own offered real estates.

Chapter 6

Sequence Diagrams

6.1 What are Sequence Diagrams?

Sequence diagrams model the logic of usage scenarios and the flow of logic within the system and therefor make it possible to document and validate the logic. Such a scenario can be a part of a use case or one or more use cases. The following figure is one of our sequence diagrams. The boxes across the top of the diagram represent classifiers or their instances, typically use cases, objects, classes, or actors. The reason why actors are a part of sequence diagrams is, because they initiate and take an active part in usage scenarios. The dashed lines hanging from the boxes represent the life span of the object during the scenario being modeled. The long, thin boxes on the lifelines are so called method-invocation boxes indicating that processing is being performed by the target object or class to fulfill a message. An X at the bottom of such a box means that an object has been removed from memory. The labeled arrows are messages between the participants of the diagrams. If the source or target of a message is an object or class the label is the name of the method invoked in response to the message.
For all sequence diagrams, please take a look at the group-paper.



6.2 Why Should You Draw Sequence Diagrams?

The reasons for drawing sequence diagrams are:

- To validate and flesh out the logic of the system.
- Great way to document the design, at least from the point-of-view of use cases.
- They help detecting bottlenecks in design

Chapter 7

Activity Diagrams

7.1 What is an Activity Diagram

Activity diagrams are used to document the logic of a single method, a single use case, of the flow of logic of a business process. The symbols used in an activity diagram are:

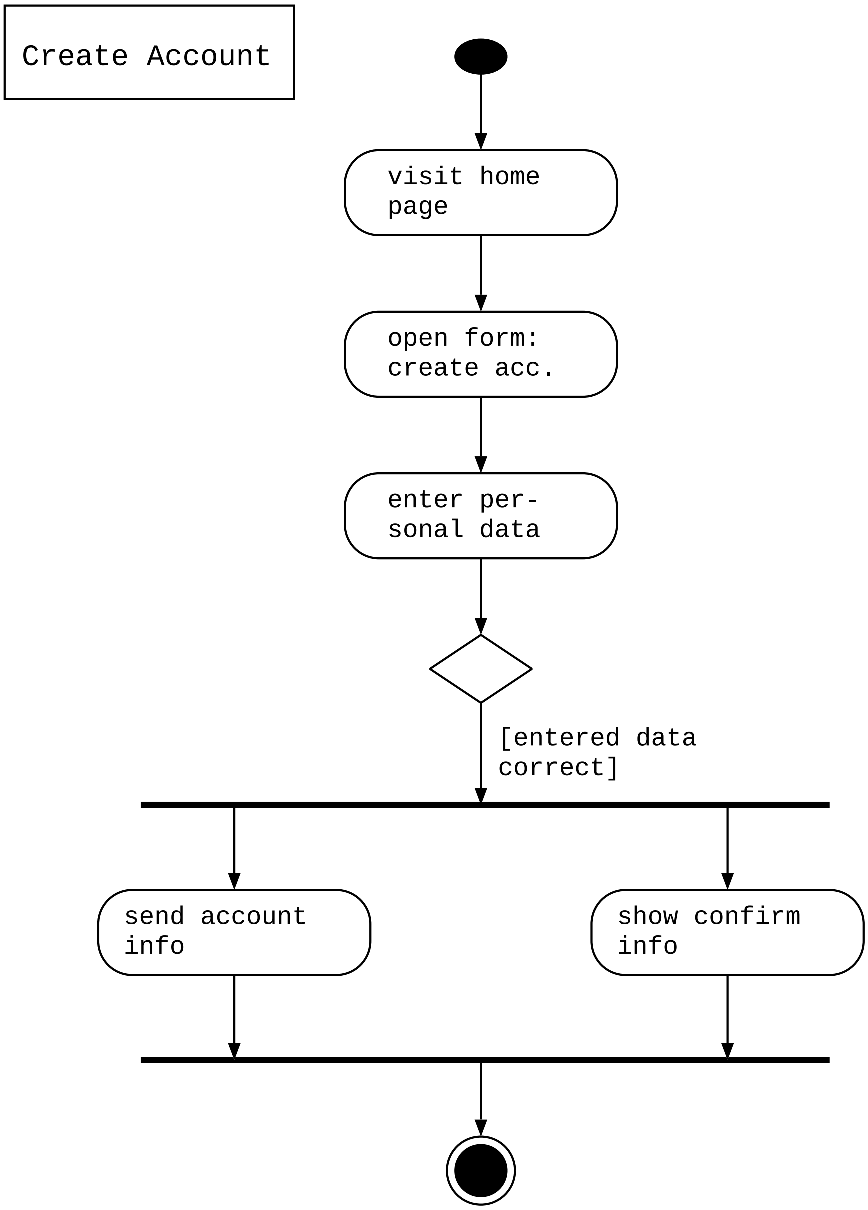
Starting/ending point: Filled circle represents the starting point and a filled circle with a border represents the ending point.

Rounded rectangles: They represent processes or activities that are performed.

Diamond: Represents a decision point, where the system or an actor decides to go one way or another.

Arrows: The arrows model the flow direction in the activity diagram. The text on the arrows represent conditions that must be fulfilled to proceed along the transition.

The thick barks: They mark the start and end of potentially parallel processes.



7.2 The Proseminar on December, 12, 2001

The group leader presented the sequence and activity diagrams we worked out.