PS Software Engineering
December 18, 2001

ImmoWarrior

Report

Andrea Geierspichler
andreageierspichler@hotmail.com

Academic Supervisor Mag. DI Dr. Roland Schwaiger

Department of Computer Science
University of Salzburg

Correspondence to:
Universität Salzburg
Institut für Computerwissenschaften und Systemanalyse
Jakob–Haringer–Straße 2
A–5020 Salzburg
Austria

ImmoWarrior

Andrea Geierspichler

Department of Computer Science
University of Salzburg
A-5020 Salzburg, Austria

Contents

Chapter 1

The Project

1.1 Introduction

In order to learn about the procedure of creating software, all groups have to act as if they were software-developers and a client (acted by the ps-leader) gives them a job. At the beginning the only information about the assignment is: the client is representative of a real estate company. The concern wants to have a system that collects and illustrate information about real estates. *Well, but who should be able to use the system?* The answer is as short as precisely: private persons as well as banks or enterprises like that. *What more particularly should the software provide?* The system should on one hand send information to clients, who don't have to install special software in order to access the data base. Thereby, it should be pretty easy for the clients to use the system. On the other hand the information should be sent to other computer systems.

Fixed date for the project is Jannuary, 2002.

1.2 About CRC Sessions

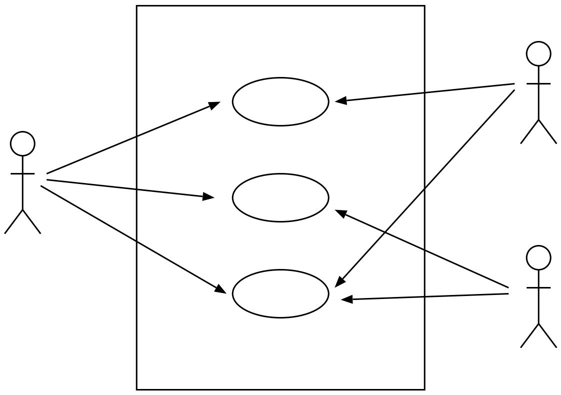
- Definition: CRC session Ward Cunningham an Kent Beck forged the CRC session in 1989. The mnemonics CRC mean: Class, Responsibility and Collaborators. *Classes* are the units that interact within the system. *Responsibility* stands for what the unique classes have to do in *collaboration* with an other class.
  In a CRC session special cards are used:

Classname	
Responsibilities	Collaborators

As you can see, classes, responsibilities and collaborators are collected on the frontside. On the backside classes and their attributes are defined more precisely.

- Procedure About 5 people should join a CRC session. If there are more than 5 people, some of them won't be involved as much as you expect. There even should be no bosses - no dominant members (!). There should be an object oriented faciliator, domain experts and developers.
  Now, what is to be done? First the group has to find out the classes and responsibilities. Therefore it is helpful to find all nouns and verbs describing the problem. The nouns are used as the classes and the verbs as responsibilities. The next step is to extract the important classes. There will be many classes/nouns that are not relevant. It is also possible, that during the development of the system the crew finds out that some more classes are not needed any more. That does not matter, they can be dropped even in later phases. Even responsibilities, sub- and superclasses can be defined at any time they become obvious. They are not needed until the design phase. Next, use cases become important:

- Definition: Use cases A unique use case defines the procedure of one scenario the system has to deal with. It is outlined that way:



The last step in a CRC session is to define these use cases. Therefore each person takes one class. One after another scenario (at the end even exceptions) is acted out and recorded.

Summarizing, a CRC session aims at defining the problem domain and representing the problem in an object oriented way.

1.3 Our CRC cards

This is the latest version of the cards we have developed during the course.

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

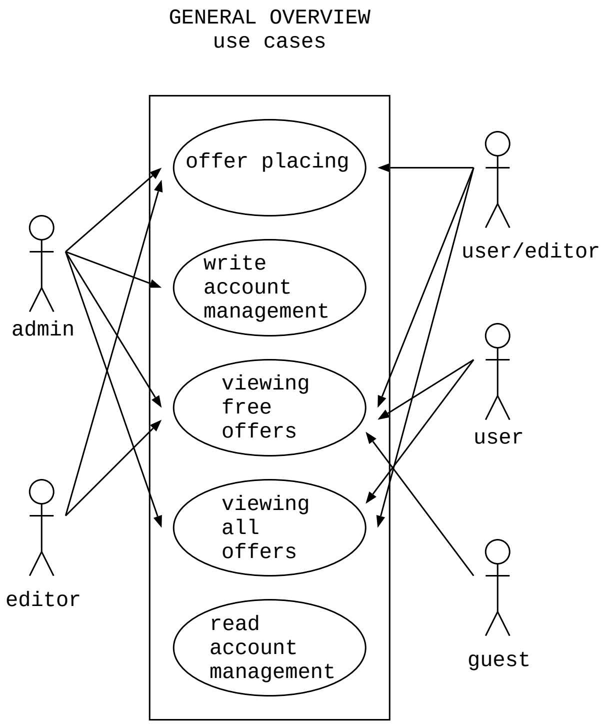
Classname: Send Real Estate Info	
Responsibilities	Collaborators
send info to e-mail/cell phone look up account info	session account

1.4 Essential use cases and essential user interface

1.4.1 Use cases

Overview

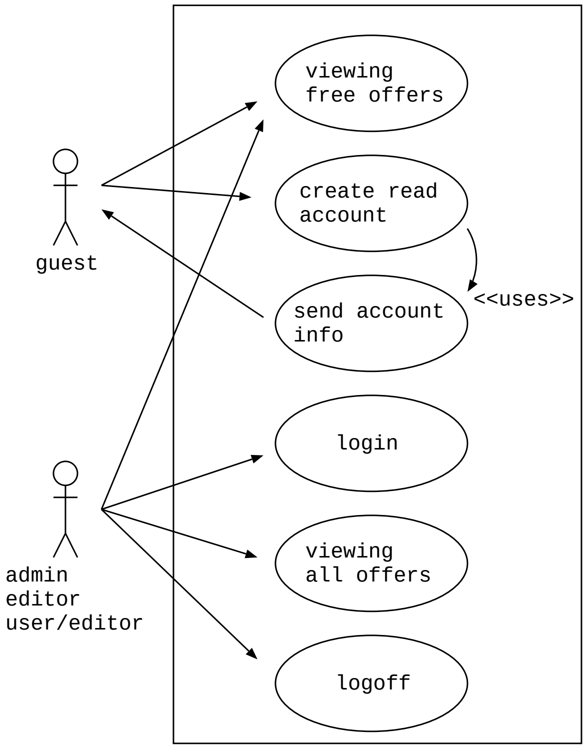
The following graphics show all use cases of our project. The first one is a general overview in which you can see the privileges of the actors.



Viewing offers, create account

The next pictures show the actions in detail: the first use cases are: viewing free offers and create read account. Viewing free offers can be carried out by everybody. Create read account is reserved to the guest. When a guest has created his read account, the system sends him back his account information containing user name and password. Then he is not a guest any longer. He has become a user. Viewing all offers can be done by 3 actors: the admin, the editor and the usereditor. Therefore they need to be logged in.

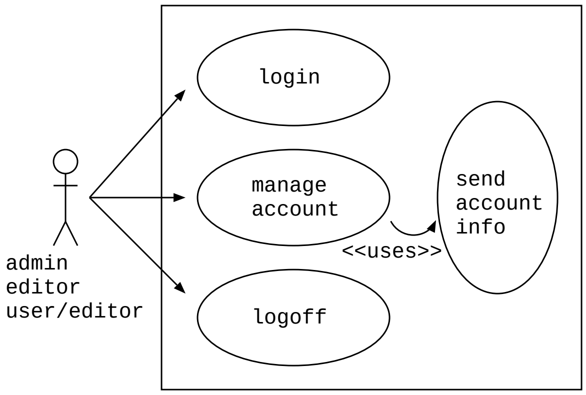
viewing free offers/create read account/viewing all offers



Manage account

The use case manage account can be acted out by the admin, who has to do all the administration work, and by the editor and the editoruser, who can change their own account. Again the system sends back the account information.

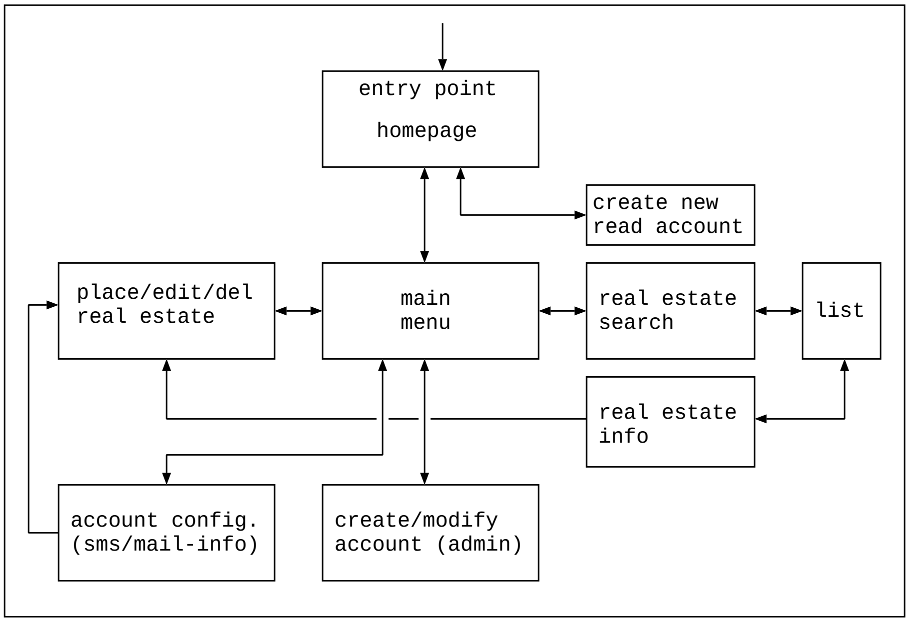
manage account



1.4.2 User interface

Overview

The following graphic shows a general overview about the user interface, followed by more detailed information.



Homepage, Main menu

When a person enters the homepage he can see about 5 free offers without being logged in. He has 2 other opportunities: logging in or, if he is not registered yet, create a new account. Then he gets into the main menu. It contains 5 links: all offers, offer placing, create account/change privileges, account configuration and logoff. According to the person's privileges he is able to click on vaious links and carry out some tasks (see page 9).

Offer placing

The "offer placing"-form is divided into some input fields and lists: you need to choose the kind of your real estate and insert location, price, description,... "Account configuration" is built up of checkboxes and input fields. You can choose whether you want information via sms or email, and insert your personal data, email adress and cell-phone number. There is also a real estate link list, through which you can be linked with your own real estate offers. You would need that to modify your offers (see page 10).

All offers

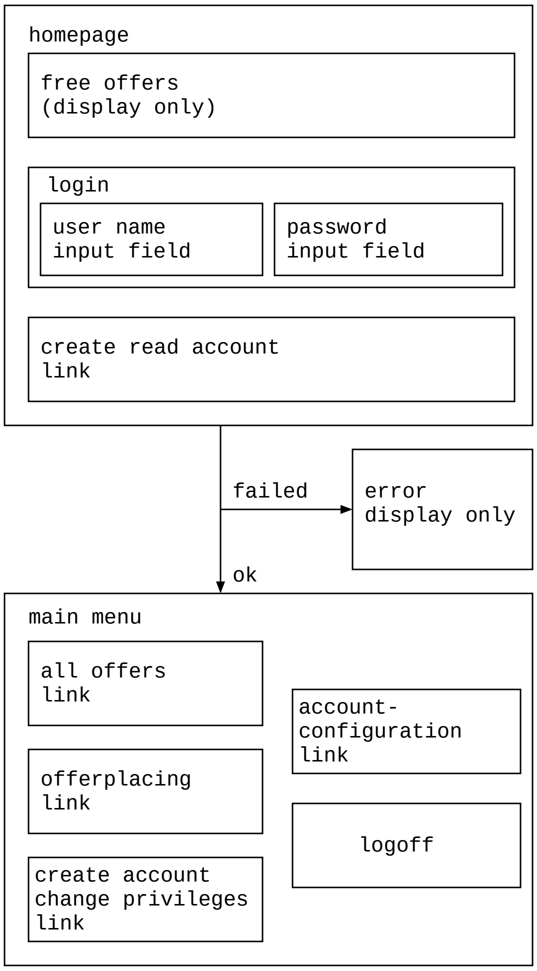
The ”all offers”-form is divided into listboxes and input fields to choose search criterias and keywords. The system will search according to them and afterwards you will be shown a link list with all the search results. By clicking on one of these links you get detailed information about that real estate (see page 11)

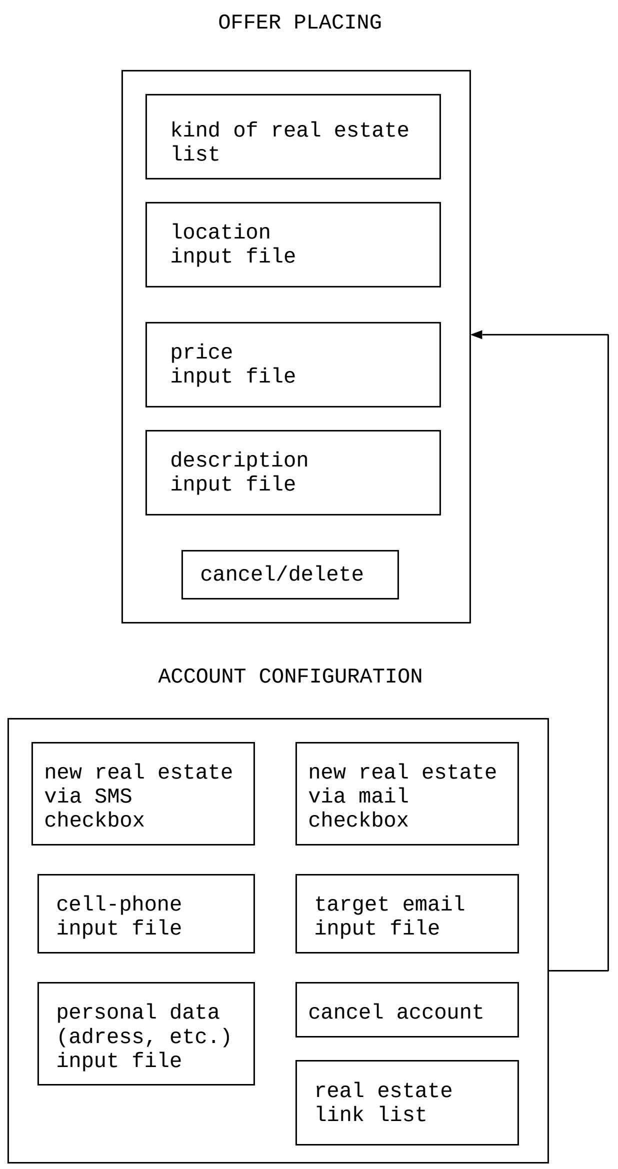
Create account, change privileges

The ”create account/change privileges” form is split up into 2 parts: one leads to an existing user, whose account can be modified. The other one creates a new account. It contains input and option fields (see page 12)

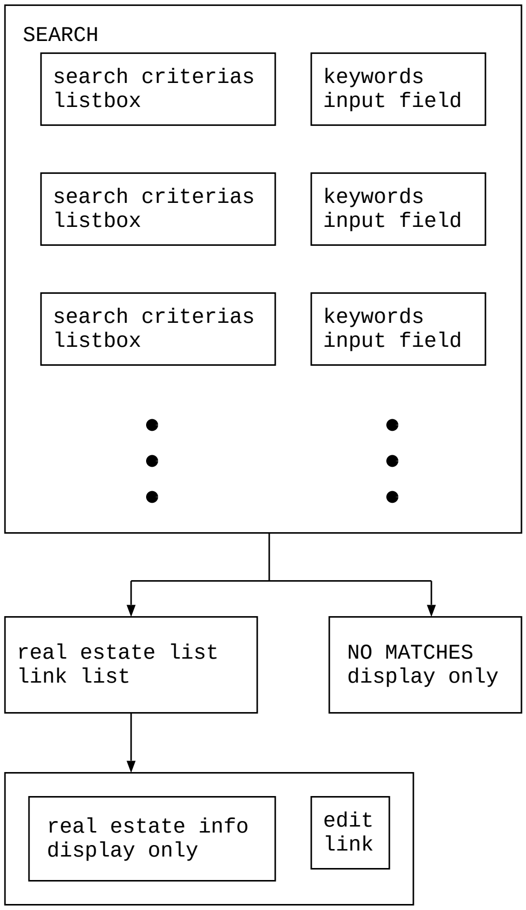
Create read account

The next graphic shows the user interface of ”create read account”. The guest has to type in title, name, adress, phone, mail, credit card number, expiring date and read the business terms (see page 12) homepage/main menu

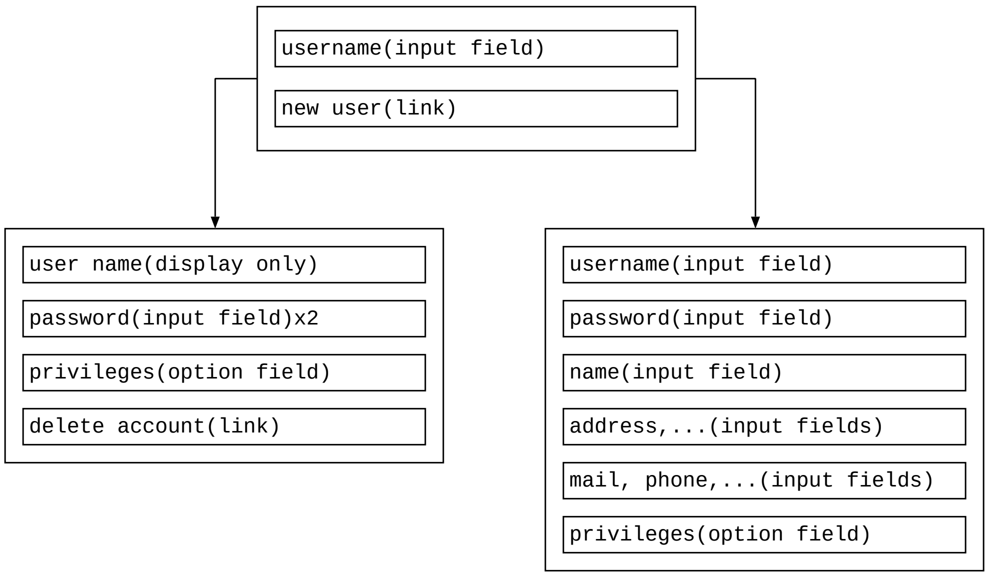



OFFER PLACING
kind of real estate list
location input file
price input file
description input file
cancel/delete
ACCOUNT CONFIGURATION
new real estate via SMS checkbox
new real estate via mail checkbox
cell-phone input file
target email input file
personal data (adress, etc.) input file
cancel account
real estate link list

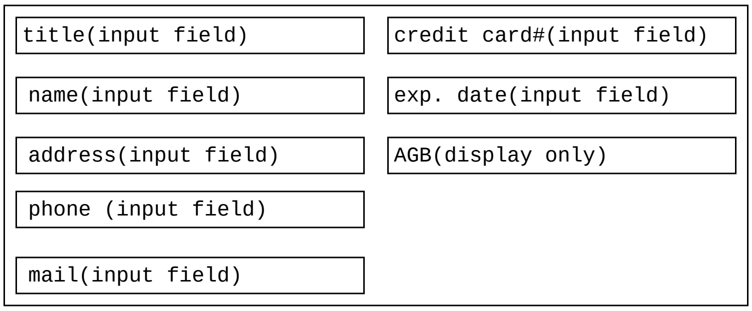
ALL OFFERS



create account/change privileges



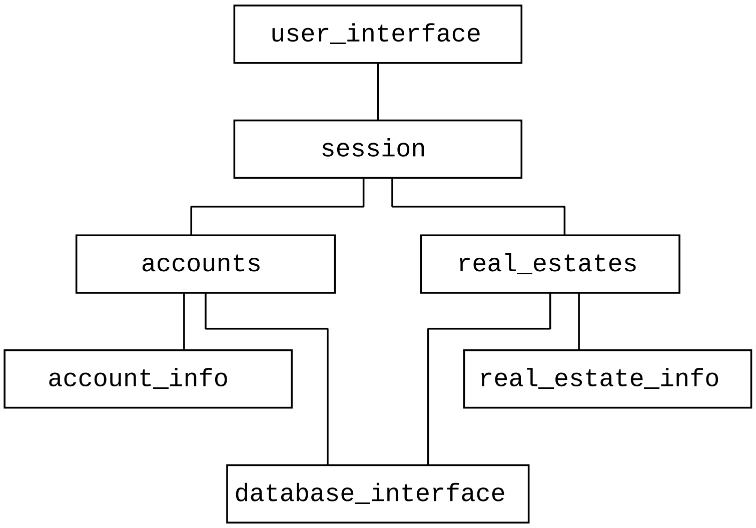
create read account



1.5 Class diagrams

The first graphic shows an overview of the class diagrams. You can see, user_interface, accounts and real_estate start a session. Accounts and real_estates also contact the database_interface. Accounts needs detailed information from account_info, whereas real_estates gets details from real_estate_info.

overview



Account controller

The account controller can access null or more account information units in order to get the required data. An account information unit itself contains user name, password, name credit card number and expiring date. It even contains one adress, one connection and one or more privileges. The only job of account information is to validate data. The account controller has to create, modify and drop accounts. Therefore it has to interact with the database interface to send and get account information (see page 14)

Real estate controller

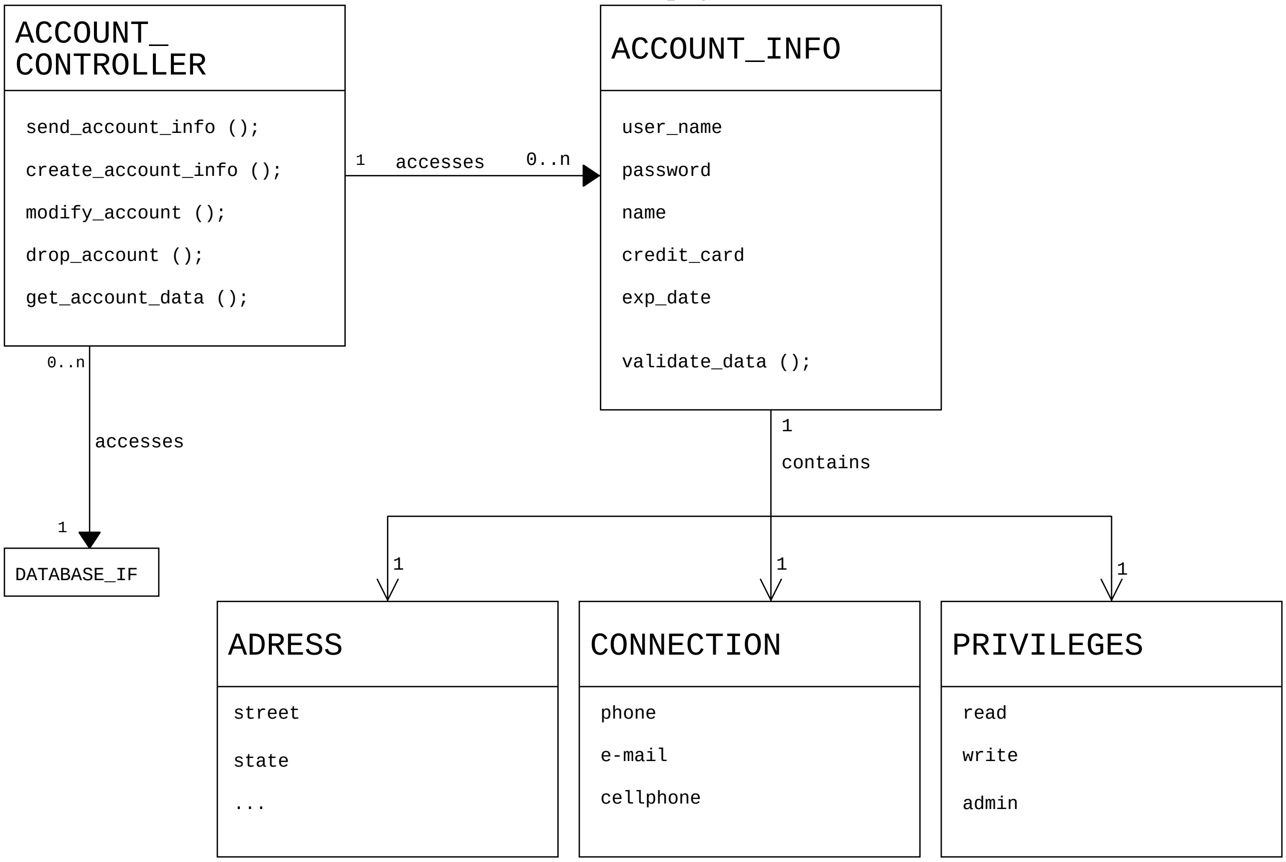
The real estate controller acts like the account conroller. It can access null or more real estate information units which contain kind, location, price and description of the real estate. The real estate information has to validate the data. The real estate controller creates, modifies, drops and searches for real estates. Therefore it sends and gets information via the database interface from the database (see page 15)
The database interface itself has to connect the real estate controller and the account controller with the database, execute sql statements and disconnect again (see page 15)

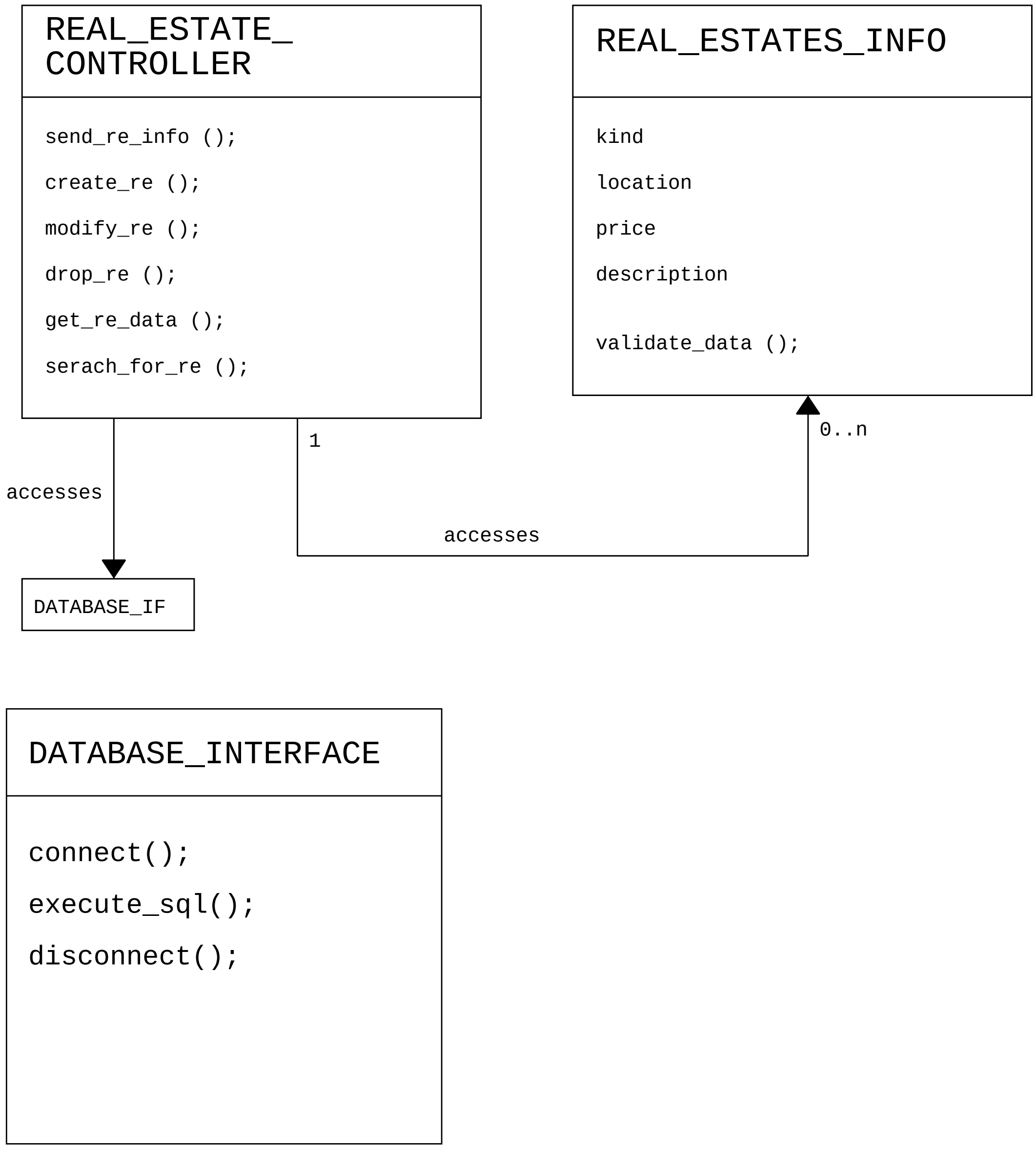
User interface

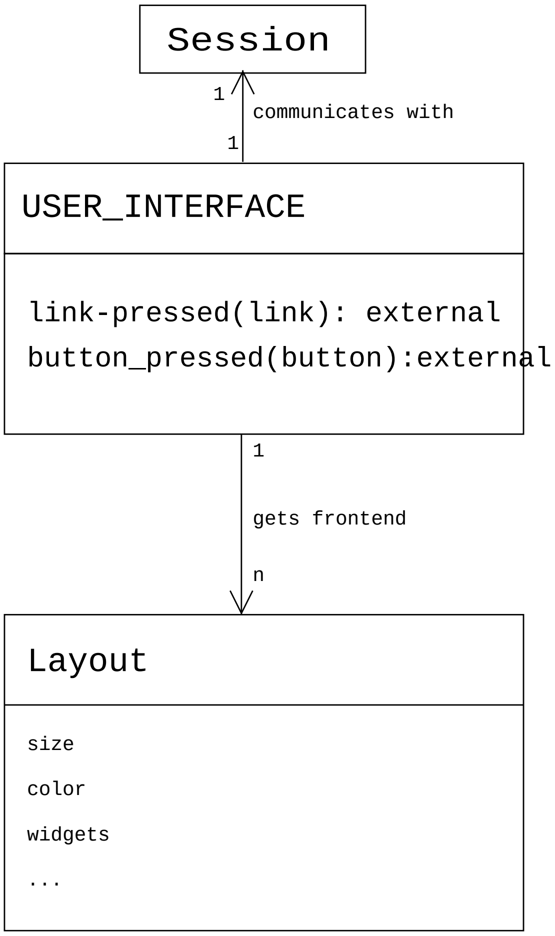
The user interface on one hand communicates with the session, on the other hand gets information about size, color, etc. from the layout. It has to tell the session which link or button has been pressed (see page 16)

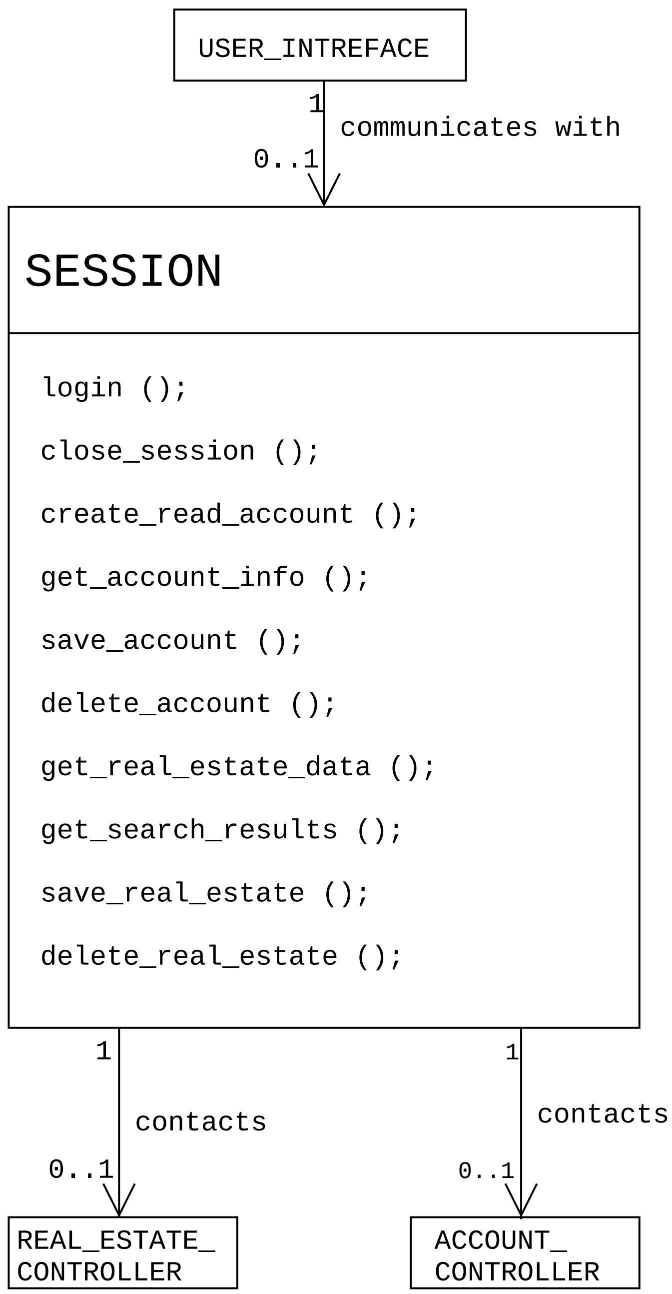
The session

The session itself communicates with the user interface. The jobs of the session are: login, logoff, create read account, get account information, save and delete account, get real estate data, get various search results, save and delete real estates. Therefore it contacts the real estate controller and the account controller (see page 17)




REAL_ESTATE_
CONTROLLER
send_re_info ();
create_re ();
modify_re ();
drop_re ();
get_re_data ();
serach_for_re ();
REAL_ESTATES_INFO
kind
location
price
description
validate_data ();
1
0..n
accesses
accesses
DATABASE_IF
DATABASE_INTERFACE
connect();
execute_sql();
disconnect();


Session
1
communicates with
1
USER_INTERFACE
link-pressed(link): external
button_pressed(button):external
1
gets frontend
n
Layout
size
color
widgets
...


USER_INTREFACE
1
communicates with
0..1
SESSION
login ();
close_session ();
create_read_account ();
get_account_info ();
save_account ();
delete_account ();
get_real_estate_data ();
get_search_results ();
save_real_estate ();
delete_real_estate ();
1
contacts
0..1
1
contacts
0..1
REAL_ESTATE_
CONTROLLER
ACCOUNT_
CONTROLLER

1.6 Sequence and activity diagrams

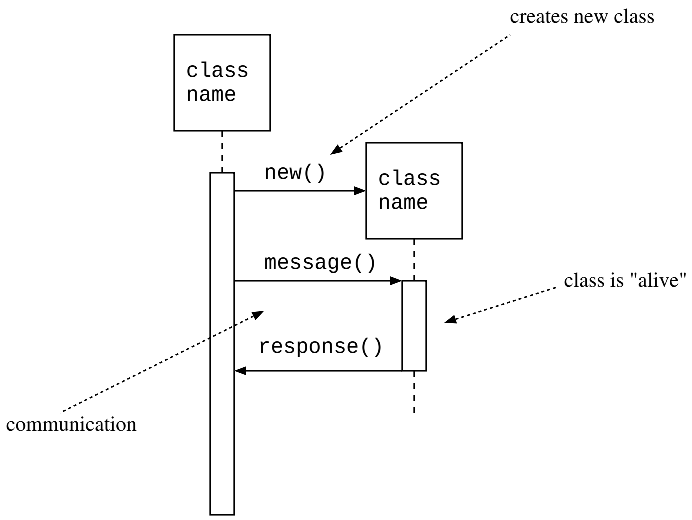
1.6.1 About sequence diagrams

Description

A sequence diagram describes the chronology of interactions between objects. The time of this sequence is limited. The interaction is just the communication between objects in order to effect a required operation. The interactions are described similar to the ones in the class diagrams, but in the sequence diagram the temporal progress of communication is of great importance.

Notation

The timeline goes upright from the top to the bottom. An object is represented by a rectangle. It is described by a vertical line, called ”lifeline” as it models the existence of the object at each point of time. If the lifeline stops you can see, that the object does not exist at that time. Messages are pictured by horizontal arrows between the object lines. They are - as the objects - inserted according to their temporal appearance.



Purpose

What do we need sequence diagrams for? It is difficult to find out the behaviour of a whole program. Sequence diagrams are simple and logical. By using them the processes and activations become much more obvious.

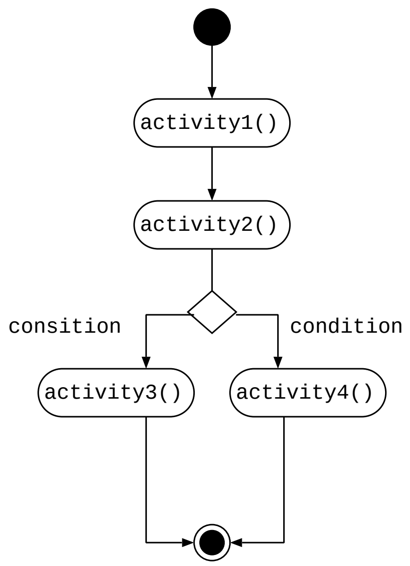
1.6.2 About activity diagrams

Description

An activity diagram describes the workflow behaviour of a program. The objects are described by their activities. An activity is a single step in a program. It is a special status that contains an internal action and one or more changes in state, the so called transitions. If an action has more transitions, they must be distinguishable from another. During the sequence of a program, an element goes through many activities. They are all visualized in the diagram.

Notation

Activities are represented by rectangles with convex rounded angles. They contain a description of the internal action. Emanating from the activity the transitions comlete the internal action and visualize the transition to the next activity. The initial state is represented by a filled circle followed by an arrow. The final state is pictured by an arrow pointing to a filled circle which lies in the middle of another one. A rhomb defines alternative paths, which are labeled with a condition.



Purpose

Activity diagrams are used to analyse the process and new concepts. They show parallel behaviour between events and activities and document decisions and iteration. All in all they help the software engineers to find new opportunities for the program.

Chapter 2

The Lessons

2.1 1^{st} Lesson - October, 25^{th}, 2001

In our first lesson we had a short introduction into the seminar and were scheduled into groups.

2.2 2^{nd} Lesson - October, 31^{st}, 2001

On the 31st of october the first meeting with the client and the first CRC session was performed. Christian Koidl carried out the duties of the groupleader, Andrea Geierspichler, Gunnar Ruhs, Roman Gross, Michael Pober were the other participants.
What the group had to do was to begin to analyse the problem domain and define the first classes.
results:

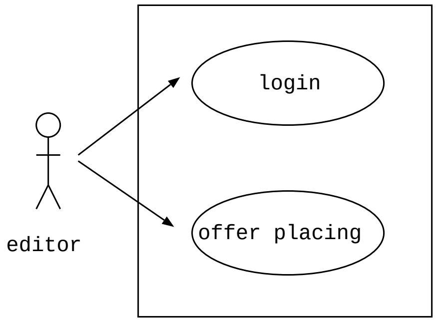
- Information about real estates can be received by the mobile phone or by the internet. Therefor there should be created a website with up-to-date information about the latest 5 real estates. If a client wants to know more about the properties or if he wants to put up a realty for sale, he needs an account. An account is not free of cost.
- Push/Pull: the system should be able to send information automatically (e.g. via sms onto the display of a mobile phone) (= push), or on individual request (= pull).
- Clients are: banks, authorities and private persons
- Clients can get just one account
- Private persons get their username/password via e-mail.
- The system should contain a unit to control access.
- Clients should be able to include multimedia-elements such as pictures, movies, sounds,... into their offer.

In the CRC session theTeam found out the following classes: Session, User Interface, Account, Permission, Information Storage, Real Estate, Editor, Private Person, Prospective Buyer. Knowledge about these classes qualified the group for the next week's job: discussion of possible use cases and thinking about the user-interface.

2.3 3^{rd} Lesson - November, 7^{th}, 2001

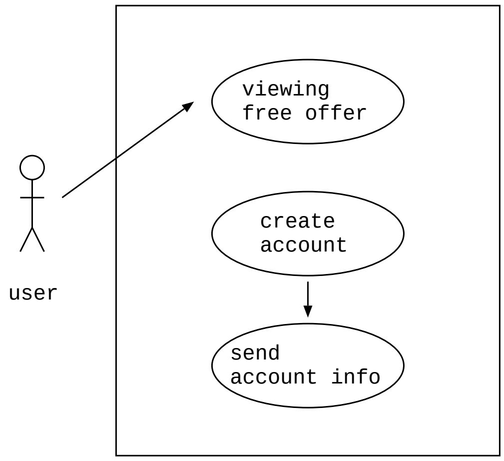
On the 7th of november theTeam met again. I was the groupleader and Christian Koidl, Gunnar Ruhs, Roman Gross, Michael Pober were the participants.
The group's job was to define the use-cases. Therefore we acted out some possible actions of the system.

- Offer-Placing



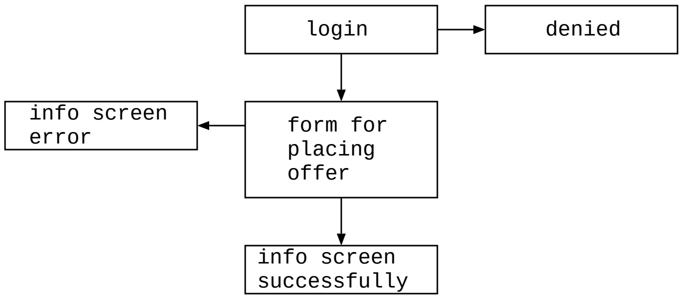
Actions: Login - Offer-Placing - Logoff
Actors: Editor
Details: The editor contacts the user-interface. The interface forwards the login-inputs to the accout, that checks in the information-storage, if the account exists. When positiv it sends the data to the privilege. The Privilege gets the user-permissions from the information-storage and returns it to the account. Then the account passes the results to the user-interface. If the user is accepted, then the interface starts a session and displays the next screen. Now the user chooses the offer-placing-menu. Here he enters his real-estate data. The interface reads the information and forwards it to the session, that stores it in the information-storage. The storage sends a success message to the user-interface over the session. The interface displays the success message and returns to the menu-screen. The user hits the logout-button and the interface closes the session and shows the login-screen again.

- Create Account



Actions: Viewing free offers - Create Account - Send Account-Information
Actor: User
Details: The User visits the Homepage. The homepage starts a session, that querries the information-storage for the free offers. The storage returns this informations via the session to the homepage and the session is closed. Now the user wants to see more details, so he enters the "create account" menu. Here a session is started and the user is offered a form, where he enters his personal information. The session gets this data and forwards it to the information-storage. Then it activates send-account-information. From here the user gets informed about his username and password. Finally the session querries for the free offers, returns them to the homepage and closes. The free-offers-screen is displayed again.

Another problem had to be solved: what should the user-interface look like?



The possible classes after the 3rd lesson were: Session, User Interface, Account, Privilege, Information Storage, Real Estate, Editor, Private Person, User, Send Account Information What our group had to prepare for the next week was a presentation of essential use cases and essential user-interface.

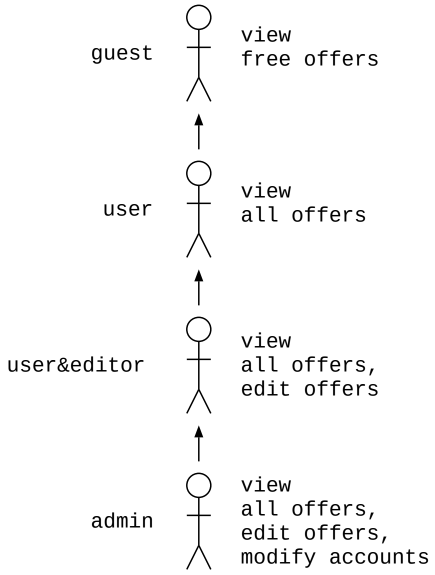
Open questions:

1. How is a user able to manage his account?
2. Is "Privileges" necessary?

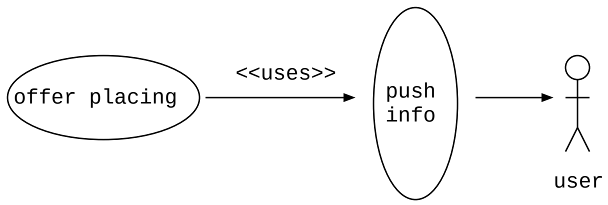
2.4 4^{th} Lesson - November, 14^{th}, 2001

On the 14th of november theTeam had to do a presentation of the essential use cases and the essential user-interface. This presentation was carried out by this week's groupleader Gunnar Ruhs.Andrea Geierspichler, Christian Koidl, Roman Gross and Michael Pober were the observing participants. Our group had prepared some slides to demonstrate the results of the previous meetings.

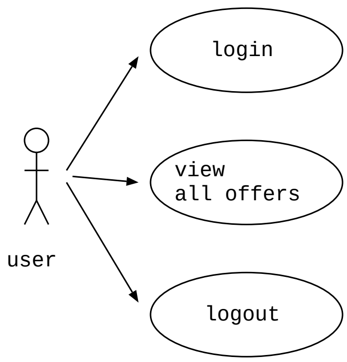
We found out that there is a inheritance between all actors, which the following picture shows:



The guest may just have a look at the free offers. The user may be logged into the system and see all offers stored in the database. The usereditor has an additional privilege: he may insert his own offers. The last one is the admin. He is granted all rights which are available: all the usereditor's and the privilege to modify accounts.
In order to push information via sms onto mobile phones, the use case offer placing needs to be modified like that:



The use case create account needs an action to send the information about accounts back to the actors.
Viewing all offers needs a precondition: the login. The picture shows the right connections:



The answer to the open question: ”What is going to happen if real estates are sold?” is: either the system has to remove it - therefore it has to check whether it was sold or not (this information has to be stored), or the editor has to remove it.
Our next week’s job was to create class diagrams to analyse the class structure.

2.5 5th Lesson - November, 21th, 2001

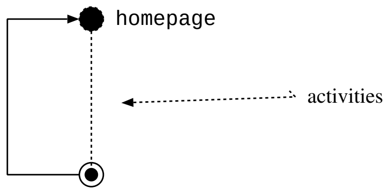
On the 21st of november the groupleader Roman Gross had to do the presentation of the class diagrams of Immowarrior. Christian Koidl, Andrea Geierspichler, Michael Pober and Gunnar Ruhs also joined the meeting.
What we found out in this lesson was, that we had to add cardinalities and names of connections to our class diagrams. Another outcome was the new class ”layout”, which we had to add.
For the following week we had to build a user interface prototype for mobile phones in WML and for the internet in HTML.

2.6 6th Lesson - December, 5th, 2001

In this lesson our groupleader Michael Pober had to present our user interface prototypes.
The crew consisting of Christian Koidl, Roman Gross, Gunnar Ruhs and Andrea Geierspichler was listening to him. You can find our HTML prototype on http://www.cosy.sbg.ac.at/m̃pober/se1/s
and the WML version on http://immowarrior.wap3.de/
One result of this meeting was, that we would have a problem if there were more than 3 privileges to be assigned. We would have to add checkboxes for coosing the required permissions. Another problem we found out was how to enable an editor to have a look at his own offers without viewing all offers. Therefore a link would be needed. In our WML version the problem occured, that the added search criterias were not listed, so that a user would have to remember them.
In the next week we had to draw sequence and activity diagrams for the program.

2.7 7th Lesson - December, 12th, 2001

This week’s groupleader Christian Koidl had to do the presentation of our sequence and activity diagrams. The auditorium consisted of Andrea Geierspichler, Roman Gross, Michael Pober and Gunnar Ruhs.
What we found out in this lesson was, that we had to add an arrow from the bottom back to homepage in the activity diagram ”create account”.



What we had to prepare for the next meeting was our personal papers.