 UNIVERSITY OF CALGARY	Course Number: SENG 609.22	Course Name: Agent-based Software Engineering
	Session: Fall, 2003	Department: Electrical and Computer Engineering
		Document Type: Project Report

# Agent-Based Electronic Realtor System

SENG 609.22 – Agent-based Software Engineering

**Project Report for Tutorials**

**Fall 2003**

**Instructor: Dr.Behrouz Homayoun Far**

*Report Prepared by:*

***NAGASHREE SASALU***

***DHANASHRI PERAMANU***

# 1. Introduction

"Realtor" it is a generic term used to describe someone who handles real estate or *realty* transactions. Realty is real property or real estate. He is an agent who buys, sells, and leases land and/or property on behalf of someone else. He can also be someone who is certified or licensed by an authority to operate an agency and having primary responsibility for transactions. In short it can be said that a realtor is a middle person who provides transaction services for a potential buyer or seller. The various transactions he provides are *search*, *coordination* and *settlement*.

*Search* is the type of transaction where in a buyer or seller tries to get information about their counterparts that best suit their expectations for buying or selling the property. *Coordination* is the effort to negotiate the price and payments. *Settlement* is the signing of legal documents, complete property transfer and money transfer.

Traditionally home or property trading is conducted through the realtor. So realtor is the middleman who helps a client to sell or buy property. **As a property seller** the client is required to select a suitable realtor who is best in his profession. The client needs to work with the realtor for property price estimation, realtor commission and advertising the sale. Also the client needs to constantly keep in touch with the realtor to know the response of the potential house buyers that visited the property. This means that event though there is a middleman, yet the homeowner has to take care of several factors that are time consuming. **As a property buyer** the client can search for the houses in newspapers, magazines, Multi Listing Systems (MLS) on Internet or taking a tour of the communities they are interested in. But this is a lot time consuming. So they can appoint a realtor who can find out the properties that are on sale that meet the client specifications. Event the buyer has to pay the realtor commission.

The whole process of property trading can be simplified from the traditional way of doing it, by having an *online realtor or realty* system. This could be a system, which can accomplish each and every stage of services for property trading. This can provide flexibility, automation of the process, around the clock and accesses to the information, negotiation and transaction capabilities for a buyer and seller with minimal commission or fees.

In this project, an electronic realtor system is presented. The objective is to develop an online system that can provide an electronic marketplace for property trading. This system should be able to provide,

- The seller with the robust facilities to list the property
- The buyer should be able to find a suitable property
- Should be able to provide advices to reduce overhead costs
- Should be able to provide the information about the companies/banks that can provide financial assistance.

## **2. System Overview**

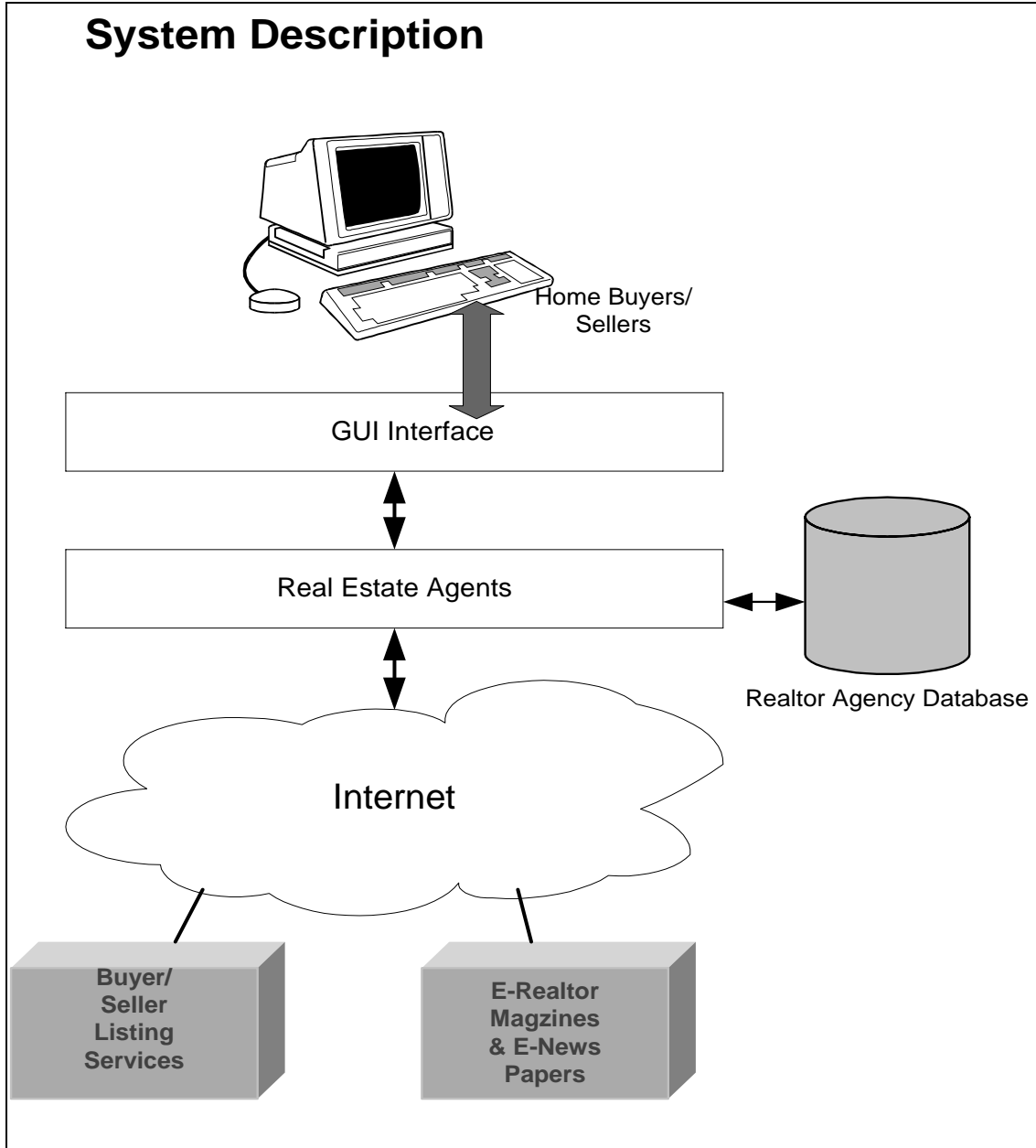
### **2.1 System Description**

The electronic realtor system should provide secure, reliable and instantaneous service simultaneously to hundreds of thousands of users. The technology should be able to considerably reduce usage of paperwork and heavy manual processes associated with the traditional real estate transaction. The system should be able to take standard documents, such as contracts between buyers and sellers, disclosure forms, and other written items, and convert them into an electronic format and should be able to transmit documents between consumers and related parties (like lawyers, banks, mortgage companies etc.) the system should be able flexible and have scalable framework in place to deal around the clock with the multiple layers of vendors and contracts involved in the typical real estate transaction.

The system should offer live help capability within a consumer's personal control center. A consumer services representative should answer questions in real time. Consumers should be able to automatically provide a feedback as they review the proposal. Buyers should also be provided with a beefed-up library of information critical for building a base of knowledge as they enter the home buying arena. The online site should allow the buyer/seller to preview what their online proposal will look like, and should be able to edit it online.

The e-realtor system should provide integrated open architecture contact, productivity and transaction applications, a loan origination and underwriting engine, MLS data, email services, digital authentications and records, all facilitated by Internet access.

Figure-1 shows the system overview for an electronic realtor system.



**Figure-1**

## 2.2 System Requirements

The requirements for an electronic realtor system at minimum is as follows:

- It should provide property listing and searching services.
- Should provide an indirect secured communication link (like email option or chat room or video conference etc.) between the buyer and seller, so that they can schedule meetings or negotiations or conduct it online.

- Should provide information about mortgage companies and banks.
- The electronic real estate magazines and electronic advertisements should be dealt with.
- It should also provide advisory services for the clients.
- The system should provide facility to connecting to other electronic home listing websites and provide prospective information.
- The system should provide all the required transaction options such as mortgage calculator, tax calculator, down payment method, billing, e-receipts etc.
- The system should support login, logout and change password transactions.
- The system should support action, search, and metadata transaction.
- The system should support buying or selling transaction updates.
- The system should provide the client with bulletin information or the notification of an email message.
- A web store should be provided for the seller, which contains property description and pictures, location, price range etc.
- A web store should be provided for the buyer that contains the buyer preferences about the property, price range and location.
- A web store should be provided for the mortgage company that contains various mortgage option package available, special offerings and services provided by the company.

### **2.3 Assumptions**

- The system is suppose to provide user-friendly Graphical user Interface which the buyer, seller or by mortgage broker can use with minimum instructions and practice.
- The system is supposed to dynamically create and maintain an agent for each user that logs in. the user can be a seller, buyer or a mortgage company. Depending on the agent created the agent information is stored.
- The system is supposed to provide information security by data encryption and use encrypted protocols.
- The database servers are assumed to be installed and setup and ready for data storage.
- The web services are assume to be established to provide search, browse and retrieve information from various Internet resources.

### **2.3 Wish List**

The following is the list of the features that could be implemented in the near future, but absent in the present system,

- Video conferencing system for potential buyer-seller pair.
- The virtual tour with instant demo from the owner about the property.

- The legal advisory services that provides the rule and regulations governing the property trading.
- The dynamic update of trade transactions happening.
- The electronic signature options for legal documents.

## 3. System Design

### 3.1 System Architecture

The system is completely online that is web-based system. This e-realtor system provides client software that meets the minimum compliance standards for communicating with a realtor Server.

#### **Transactions**

The *login transaction* is the first transaction that a client must issue during a session with a host. It identifies to the host the username and password of the human user, giving the host the opportunity to reject the login request or apply business rules as needed. The login transaction uses a simple challenge-response method called Digest Authentication in order to prevent sending the user's password over the Internet in clear text form, providing a minimal level of security. In the response, the host provides the client software with a number of valuable pieces of information, the most important of which is a list of URLs that give the client access to the various transactions and other capabilities of the host. Besides the standard (required) transactions, the host can optionally provide additional URL links for additional capabilities, which help to ensure that the standard is not only extensible in future standards, but is also extensible by private agreement.

The *login complete* transaction is the second part of the login transaction that enables software copy protection security/piracy prevention. It allows the client and host to synchronize the information kept to prevent piracy.

The *action transaction* is used to provide the client with bulletin information or the notification of an email message. The client application should provide a means for the user to view the retrieved document. This transaction must be performed immediately after the login transaction.

The *object transaction* is used to retrieve additional information that hosts store related to specific data objects. This transaction is used primarily to retrieve multimedia types associated with host records (house pictures, sound clips, 3D tours, etc.). It also allows the client to download the latest copy of the metadata, thereby enabling the client to configure itself to the host and adapt to changes.

The *logout transaction* terminates a particular client/host session. In effect, the Logout transaction tells the host that the client has completed all processing that is going to be done in this session, and allows the host to free up any information it had been keeping on behalf of the client. It also ensures that the host logs the user out of the system, and for

systems that prevent simultaneous login of the same username, ensures that the username can log in again in the future.

The *search transaction* enables a client to request a set of data from the host, and gives the host a number of additional parameters with which to work. These parameters include things like the type of data to be returned, the query itself, the data fields to return and the format for the data returned. The host executes the query and returns the data or an error code to the client for further processing.

The *get transaction* simply allows the client to get an arbitrary file from a specific URL on the host. This transaction might be used to acquire additional host information or to download client software updates.

The *change Password* transaction provides a means for the user to change their password. The new password is appended to the username and encrypted using the Data Encryption Standard.

The *update transaction* is used to modify data on the server. The client transmits information describing the update to perform. The server then validates the information. If there are errors in the data, the server returns an error reply. If there are no errors, the record as it was inserted/updated on the server would be returned. The record is returned in the same manner as a record is returned from a search.

The *get metadata* transaction is used to retrieve structured information known as metadata related to the system entities. Metadata requested and returned from this transaction are requested and returned as MIME media types.

## **Agents**

Figure-2 shows the electronic realtor system architecture.

The Login Agent is required to differentiate different types of users depending on the feature that they are buyer, seller or mortgage company, provide authentication to a valid user and create a new user account.

The buyer, seller and banker agents are required to perform various related functionalities for which they are created.

The negotiation agent is required to provide consultation regarding the price negotiation, mortgage package and online meeting scheduling between buyer, seller and banker.

The transaction agent keeps track of various transactions that are happening in the system as explained in *section: Transactions*.

The Buyer/Seller database agent is required to create, update, manage, modify, delete or retrieve appropriate information in the data base server.

The Marketing agent is required to create, update, modify, delete or retrieve the information that requires to be used to publish on electronic magazines and newspapers and dynamically updates the web resources.

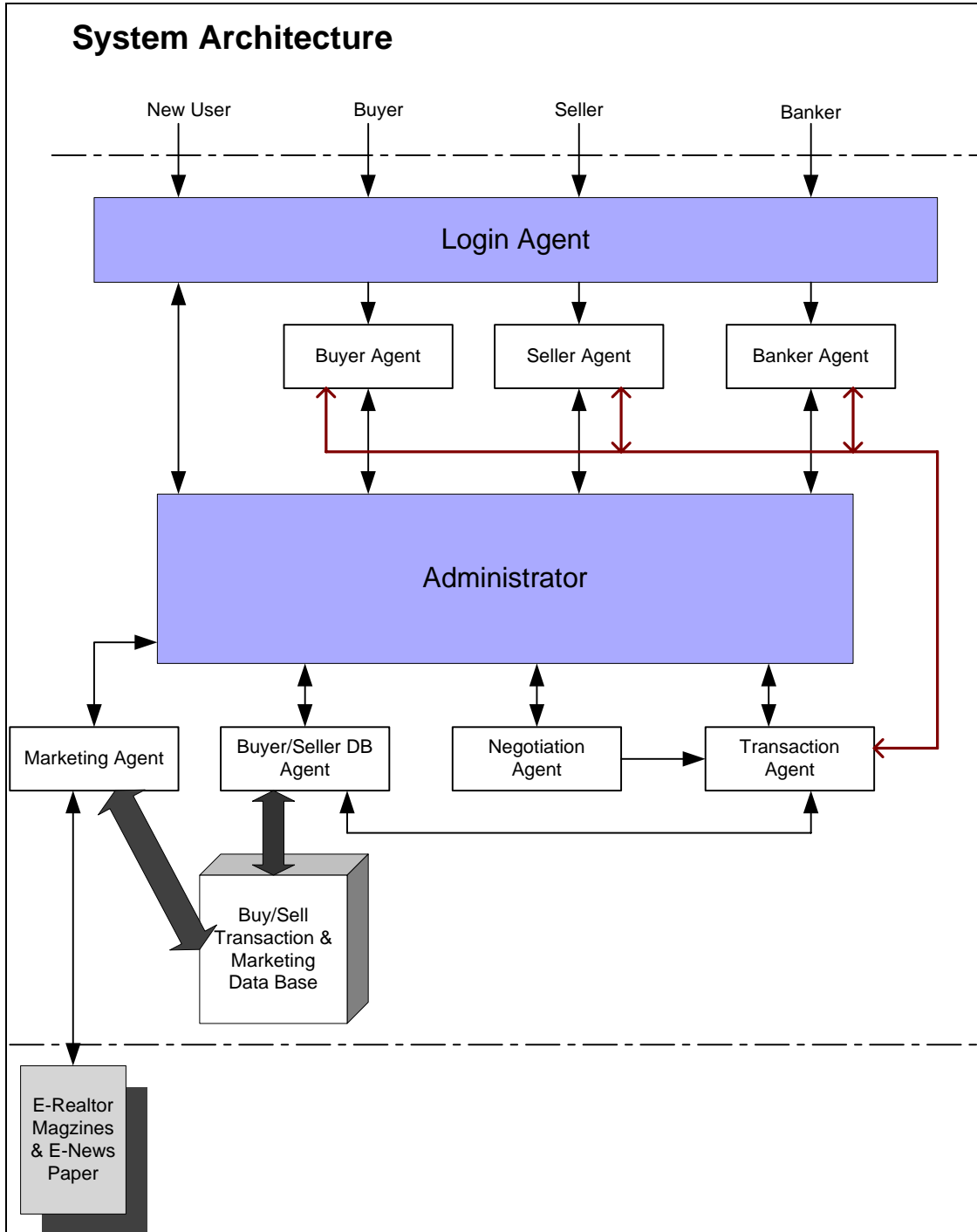


Figure-2

### **3.2 Role Identification**

The following roles have been identified in the electronic realtor system.

- An agent for home buyer
- An agent for home seller
- An agent for mortgage company or banker
- An agent for logging procedure
- An agent for negotiation
- An agent for transaction
- An agent to work with the system database
- An agent to coordinate the administration work

### **3.2 Agent Description**

This section provides a brief description about each agent that has been identified in the electronic realtor system.

#### **Login Agent**

The login agent authenticates the user depending on the type of the user that is logging in. If it is a new user, then it requests the administrator to create a new account for the user in the buyer/seller database. Depending on the type of the user, the Login agent provides complete, partial or limited access to the system.

The Login agent is responsible to verify the new user account information provided the first time, and depending on the validity of the data has the power to accept or reject the login request. It is also responsible to crosscheck the login-id and password each time the user tries to login. It is also responsible for requesting the administrator to check the buyer/seller database to delete and notify the inactive or expiring user accounts.

#### **Buyer Agent**

The buyer agent provides the property information query depending on the house description, price and other information provided by the buyer and retrieves all the matching listings that are available in the system database through the Buyer/Seller database agent.

It can also request the seller to provide a virtual tour of the house. It can also schedule meeting with seller and banker through the negotiation agent. It also participates in the final transaction, which is the purchasing of the house and transfer of property through the Transaction agent.

The Buyer agent interacts with Buyer/Seller DB agent, Negotiation agent and Transaction agent.

**Seller Agent**

The seller agent provides all the services for the user to create and post information about the house for sale. This agent request the administrator to update the database with the information provided by the seller through the Buyer/Seller agent.

It can also schedule meeting with buyer and banker through the negotiation agent. It also participates in the final transaction, which is the selling of the house and transfer of property through the Transaction agent.

The Seller agent interacts with Buyer/Seller DB agent, Negotiation agent and Transaction agent.

**Banker Agent**

The banker agent allows various mortgage companies and banks to post information about various mortgage packages and special offers available for the buyers and sellers. This agent request the administrator to update the database with the information provided by the banker through the Buyer/Seller database agent.

The banker agent also participates in the negotiation with the buyer and seller. It also participates in the final transaction of selling or buying the house. It can also provide the buyer or seller with the advice to select proper mortgage.

The Seller agent interacts with Buyer/Seller DB agent, Negotiation agent and Transaction agent.

**Negotiation Agent**

The negotiation agent provides a platform for the buyer, seller and the banker to interact with each other to negotiate the price and mortgage rate.

This agent will send all activity information to the transaction agent, which will request the Buyer/Seller DB agent to store in the database.

It interacts with Buyer agent, Seller Agent, Banker agent and Transaction agent.

**Transaction Agent**

The transaction agent interacts with Buyer/Seller DB agent to update every type of transactions that are happening in the system.

It interacts with Buyer/Seller DB agent and Transaction agent.

**Buyer/Seller Database Agent**

This agent provides all database functionalities. It updates, modifies, deletes, stores and retrieves the information in the system database depending on various other agent requests.

It interacts with Buyer agent, Seller agent, Banker agent, Buyer/Seller DB agent and Transaction agent.

### **Administrator**

The administrator is the coordinator, which provides a proper gateway for various activities happening in the system. It is responsible for the security of the system and collaborates various communications.

It is a central controller who coordinates, synchronize and guard the whole system.

### **3.3 Agent Internal Architecture**

Each agent is an entity, which as various important components that work together to provide complete dynamic nature of a software agent.

Figure-3 shows the internal architecture of an agent created by the electronic realtor system.

The agent has an internal database wherein the agent profile is stored. The Listener/Interpreter provides the interface link to interpret the incoming data and outgoing data. The processor parses the data and processes the information. The processor also has a service component that provides any external information that might be required by the processor during information processing. This information can be from other agents or database or web resources.

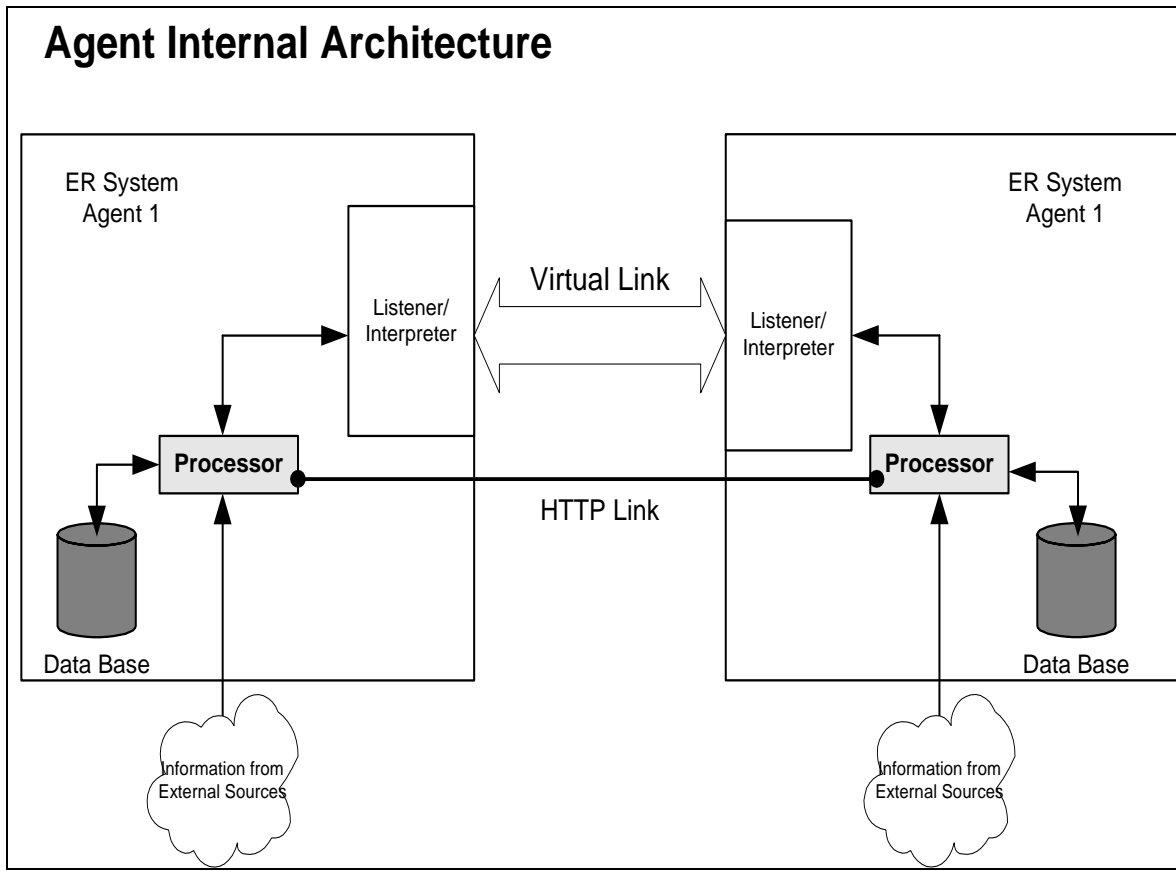


Figure-3

## 4. Technology Overview

### UNICOREpro

The Grid solution package **UNICOREpro** (professional Uniform Interface to Computing Resources) offers customers a **ready-to-run Grid system** including client and server software. **UNICOREpro** makes distributed computing and data resources available in a **seamless and secure way** through intranets and Internet.

The **UNICOREpro** client provides a graphical interface for preparing jobs, submitting them to remote computing resources, and monitoring their progress. The Pallas Grid solution package **UNICOREpro** consists of

- The industrial-quality implementation of the UNICORE system.
- Administrator support for installation and maintenance.
- User support and training.
- Consulting services for analysis and choice of the right Grid configuration.

## **DataMirror**

DataMirror Transformation Server is a high performance, peer-to-peer data integration solution that saves businesses time and resources by providing programming-free data integration across all computers in an enterprise. Using Transformation Server's unique capture, transform and flow (CTF) technology, companies can share vital information with customers, partners and employees, allowing them to make intelligent business decisions, improve customer service levels and boost revenues.

Transformation Server integrates data bi-directionally and in real-time between DB2 UDB, Microsoft SQL Server, Oracle, Sybase and XML across multiple computing platforms. Transformation Server's out-of-the-box support for leading databases makes it an ideal solution for a range of distributed data applications including enterprise application integration, e-Business, business intelligence and customer relationship management.

Transformation Server's graphical Enterprise Administrator Monitor tool allows companies to gain heightened levels of system management for their integration network environment. Any integration network, whether simple or complex, can be easily visualized through user-defined diagrams. These diagrams provide an intuitive method for assessing the current health of a system, allowing users to quickly determine the status and latency of replication at a detailed level. Transformation Server allows companies to conduct business in the real-time economy, reduce their total cost of ownership (TCO) and leverage the technology assets they already have.

## **SOAP - The Key Behind Web Services**

SOAP stands for Simple Object Access Protocol. It is one of the key technologies that enable web services to provide the benefits described above. In order for heterogeneous web service clients and servers to properly communicate with each other, they need a common protocol - SOAP.

In the real world, when two parties speaking different languages wish to communicate, they use an interpreter. This interpreter is familiar with both languages and translates one directly into the other. In the technological world, this paradigm would be infeasible. There are too many possible combinations of language choices to write direct translators for them all. Instead, SOAP is used. It can be thought of as a "universal translator" that can be converted from and to any programming language. (At least, any language that has a SOAP library for it - the list keeps growing.) Clients submit requests using their native programming language. These requests get translated into SOAP and passed over the network. Web service servers receive these SOAP messages and translate them back into something the individual web services can understand. The same process happens in the other direction when web services need to communicate back with the clients.

SOAP is built using XML. (eXtensible Markup Language). This means that, by definition, any valid SOAP transaction is a valid XML document. We do not attempt to go into detail about the specifications of SOAP in this paper.

## **CORBA**

CORBA specifies a system that provides interoperability between objects in a heterogeneous, distributed environment and in a way transparent to the programmer. Its design is based on OMG Object Model.

The OMG Object Model defines common object semantics for specifying the externally visible characteristics of objects in a standard and implementation-independent way. In this model *clients* request services from *objects* (which will also be called servers) through a well-defined interface. This interface is specified in OMG IDL (*Interface Definition Language*). A client accesses an object by issuing a *request* to the object. The request is an event, and it carries information including an operation, the *object reference* of the service provider, and actual parameters (if any). The object reference is an object name that defines an object reliably.

The central component of CORBA is the *Object Request Broker (ORB)*. It encompasses the entire communication infrastructure necessary to identify and locate objects, handle connection management and deliver data. In general, the ORB is not required to be a single component; it is simply defined by its interfaces. The ORB Core is the most crucial part of the Object Request Broker; it is responsible for communication of requests.

The communication between the object implementation and the ORB core is effected by the *Object Adapter (OA)*. It handles services such as generation and interpretation of object references, method invocation, security of interactions, object and implementation activation and deactivation, mapping references corresponding to object implementations and registration of implementations. It is expected that there will be many different special-purpose object adapters to fulfill the needs of specific systems (for example databases).

OMG specifies four policies in which the OA may handle object implementation activation: *Shared Server Policy*, in which multiple objects may be implemented in the same program, *Unshared Server Policy*, *Server-per-Method Policy*, in which a new server is started each time a request is received, and *Persistent Server Policy*. Only in the Persistent Server Policy is the object's implementation supposed to be constantly active (if it is not, a system exception results). If a request is invoked under any other policy the object will be activated by the OA in the policy specific way. In order to be able to do that, the OA needs to have access to information about the object's location and operating environment. The database containing this information is called *Implementation Repository* and is a standard component of the CORBA architecture. The information is obtained from there by the OA at object activation. The Implementation Repository may also contain other information pertaining to the implementation of servers, such as debugging, version and administrative information.

Figure-4 shows the communication model of the electronic realtor system.

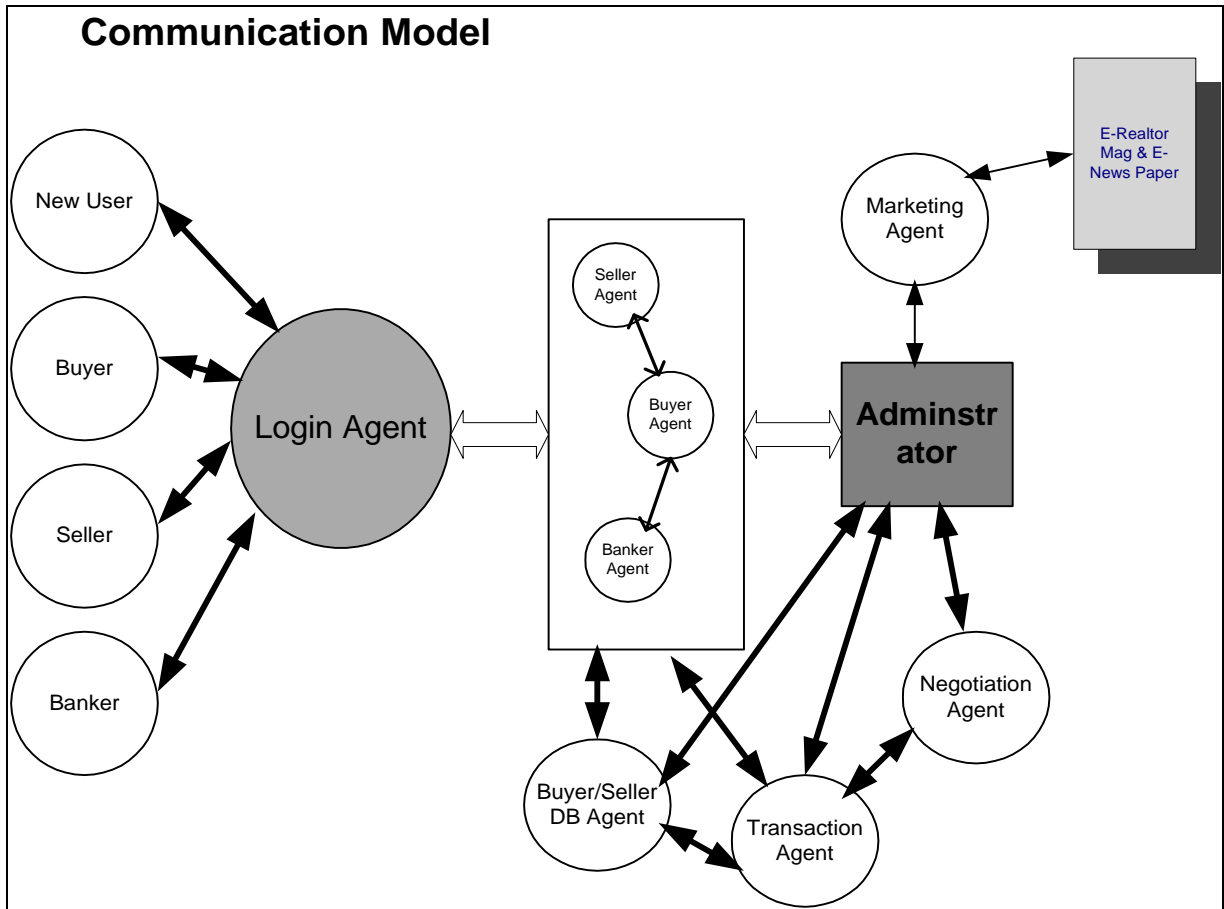
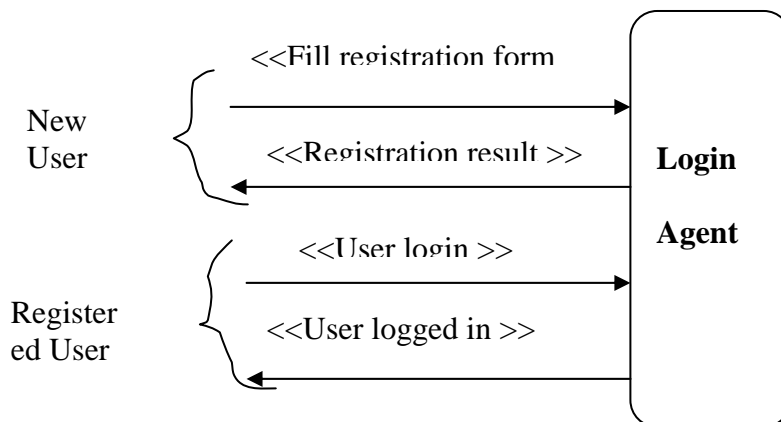


Figure-4

### 5. Detailed Design

This section lists all the use cases and description for all the participating agents in the electronic realtor system.

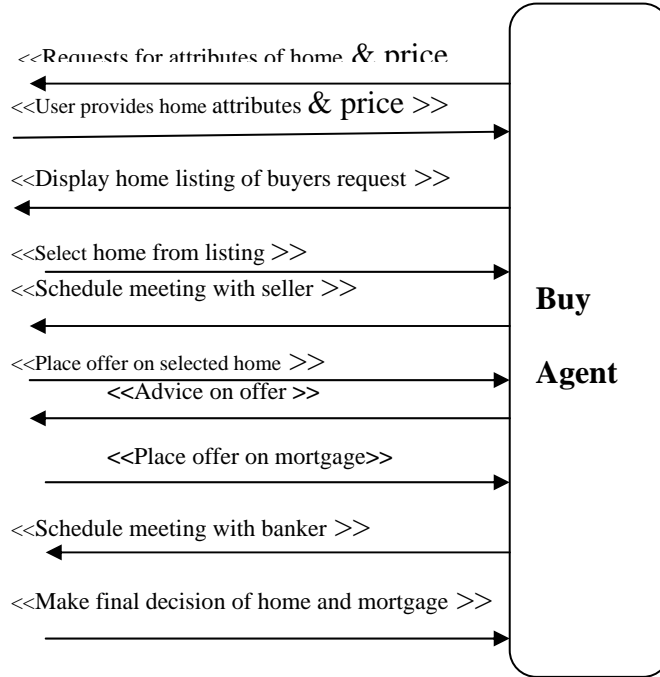
#### Login Agent:



**Use case definition:**

<b>Brief description</b>	Buyers or Sellers uses this use case to register in the system, and to identify the registered agents.
<b>Description</b>	User has register
<b>Process steps (New User)</b>	
1	New user requests to use the system. New user has to fill registration form.
2	This registration form set to Administration agent who decides to accept or reject new user or requests more information from the user.
3	If Administrator Agents response is positive then user gets connected to the system.
<b>Process steps (Registered User)</b>	
1	User has to enter login id and password
2	Login Agent checks registry, and if user is registered, Login Agent requests Administrator Agent to look up for the User Agent to connect the user to the system.
3	User Agent gets connected and can start to communicate with the system directly.
<b>Exceptions</b>	
1	Error message generation saying system is not available at this moment. Process termites.
<b>Relationship</b>	
<b>Initiating</b>	Login Agent
<b>Collaborating</b>	Administrator Agent
<b>Data Requirements</b>	
1	User Information
2	Financial status (Buyer and Banker)
3	Credit history and criminal record (Buyer)

**Buy Agent:**

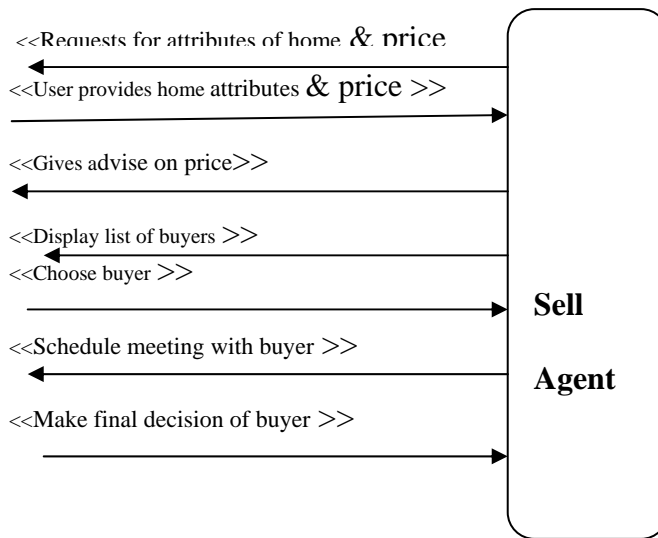


**Use case definition:**

<b>Brief description</b>	Buyer uses this use case. Buyer Agent provides all kind of services to the user such as do home searching, schedule meeting with seller or banker etc.
<b>Description</b>	Buyer must have to register
<b>Process steps</b>	
1	Buyer Agent requests for attributes of home and price range.
2	Buyer provides the information required.
3	Buyer Agent requests to Administrator Agent to provide home listing of users request
4	Buyer Agent displays the list of homes as per buyers range
5	Buyer chooses home
6	Buyer Agent contacts Sell Agent and schedules a meeting with buyer
7	Buyer places offer on selected home
8	Buyer Agent contacts Negotiation Agent regarding buyers offer and it advises buyer on offer
9	Buyer adjusts offer accordingly
10	Buyer Agent contacts Administrator Agent regarding buyers offer. Administrator Agent contacts Banker Agent to find out what best deals it has. Administrator Agent lists mortgage companies.
11	Buyer places offer on mortgage

12	Buyer Agent contacts Banker Agent and schedules a meeting with buyer
13	Buyer makes final choice of home and mortgage company
14	Buy agent informs Administrator Agent that deal is settled. Administrator Agent stores the transaction in database and destroys Buy agent.
<b>Exceptions</b>	
1	Error message is generated if database is not accessible
2	“No house is found” message is generated if no house is found as per buyers requirement
<b>Relationship</b>	
<b>Initiating</b>	Buyer Agent
<b>Collaborating</b>	Administrator Agent, Data Base Agent, Negotiation Agent, Banker Agent, Sell Agent
<b>Data Requirements</b>	
1	Home information and price range
2	Seller and Banker information

**Sell Agent:**

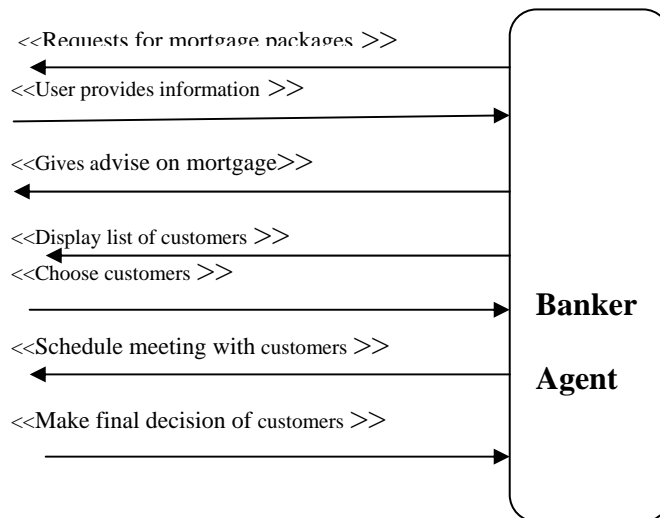


**Use case definition:**

<b>Brief description</b>	Seller uses this use case. Sell Agent provides all kind of services to the seller such as schedule meeting with buyer, get advise on home selling etc.
<b>Description</b>	Seller must have to register

<b>Process steps</b>	
1	Sell Agent requests for attributes of home and price range.
2	Seller provides the information required.
3	Sell Agent requests to Administrator Agent to provide listing of buyers
4	Sell Agent contacts Negotiation Agent regarding Seller offer and it advises buyer
5	Seller Agent displays the list of buyers
6	Sell Agent chooses buyer
7	Sell Agent contacts Buy Agent and schedules a meeting with buyer
8	Sell Agent contacts corresponding buyer, if buyers are interested, Sell and Buy Agent schedules meeting
9	Buyer places offer on mortgage
10	Seller makes final choice on buyer
11	Sell agent informs Administrator Agent that deal is settled. Administrator Agent stores the transaction in database and destroys Sell agent.
<b>Exceptions</b>	
1	Error message is generated if database is not accessible
2	“No house is found” message is generated if no house is found as per buyers requirement
<b>Relationship</b>	
<b>Initiating</b>	Sell Agent
<b>Collaborating</b>	Administrator Agent, Data Base Agent, Negotiation Agent, Buy Agent
<b>Data Requirements</b>	
1	Home information and price range
2	Seller and Banker information

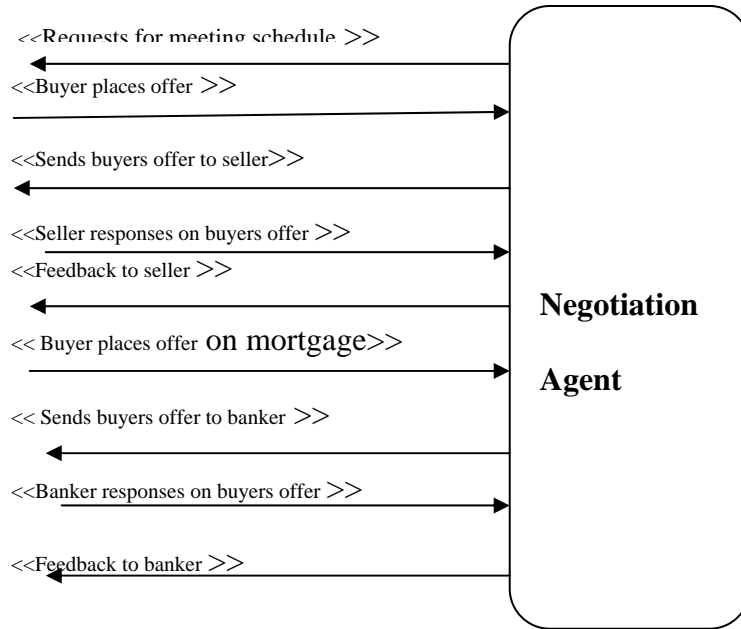
**Banker Agent:**



**Use case definition:**

<b>Brief description</b>	Mortgage company uses this use case. Banker Agent provides all kind of services to the seller such as schedule meeting with customer, get advise on mortgage etc.
<b>Description</b>	Banker must have to register
<b>Process steps</b>	
1	Banker Agent requests for mortgage package available.
2	User provides required information.
3	Banker Agent displays advice and comments on mortgage.
4	Banker Agent contacts Administrator Agent for list of buyers who need mortgage package similar to one its client provided
5	Banker Agent displays the list of customers
6	User chooses customer to contact from the list
7	Banker Agent contacts Buy Agent and schedules a meeting with buyer
8	Banker Agent contacts corresponding buyer, if buyers are interested, Banker and Buy Agent schedules meeting
9	User chooses customer
10	Banker Agent informs Administrator Agent that deal is settled. Administrator Agent stores the transaction in database and destroys Banker agent.
<b>Exceptions</b>	
1	Error message is generated if database is not accessible
2	“No house is found” message is generated if no house is found as per buyers requirement
<b>Relationship</b>	
<b>Initiating</b>	Banker Agent
<b>Collaborating</b>	Administrator Agent, Data Base Agent, Buy Agent
<b>Data Requirements</b>	
1	Mortgage information

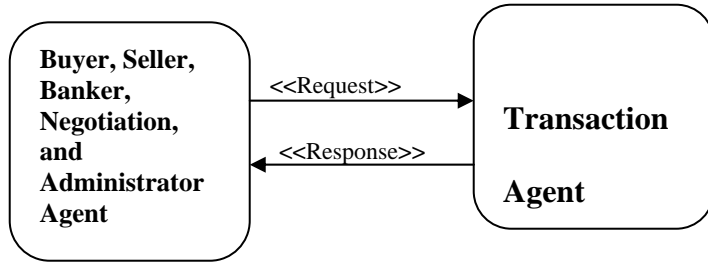
**Negotiation Agent:**



**Use case definition:**

<b>Brief description</b>	Negotiation Agent provides all kind of services to the Buyers, Sellers and Banker Agents to communicate with each other
<b>Pre Condition</b>	Negotiation Agent will be always in waiting state to get signal from Buyer, Seller, and Banker
<b>Process steps</b>	
1	Negotiation Agent requests for meeting schedule from buyer to seller and banker. Negotiation Agent waits for the response and then feeds back
2	Negotiation Agent gets house offer from buyer, Negotiation Agent sends this offer to Seller and waits for its response and once he gets response then he feeds back this response
3	Negotiation Agent gets mortgage offer from buyer, sends this offer to Banker and waits for its response and once he gets response then he feeds back this response
4	Negotiation Agent sends this record to Transaction Agent
<b>Exceptions</b>	
1	Negotiation failed error message is generated if he doesn't get any response from other side
<b>Relationship</b>	
<b>Initiating</b>	Banker Agent, Buyer Agent, and Seller Agent
<b>Collaborating</b>	Banker Agent, Buyer Agent, and Seller Agent, and Transaction Agent
<b>Data Requirements</b>	
<b>Data Required</b>	Time, place and price offer

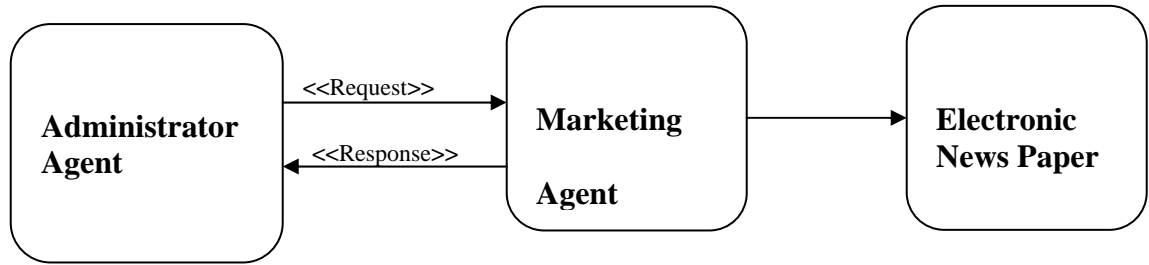
**Transaction Agent:**



**Use case definition:**

<b>Brief description</b>	Transaction Agent keeps track of all transaction processes
<b>Pre Condition</b>	Connection with Negotiation and Buy/Sell DB Agent
<b>Process steps</b>	
1	Any negotiation between Buyer, Seller and Banker must pass through Transaction Agent
2	As one transaction starts Transaction Agent tells Buy/Sell DB Agent to open temporal table in Buy/Sell DB to record transaction process
3	This temporary table must be updated dynamically on every transaction process
4	As one transaction ends Transaction Agent tells Buy/Sell DB Agent to delete temporal table
<b>Exceptions</b>	
1	Error message is generated if database is not accessible
<b>Relationship</b>	
<b>Initiating</b>	Negotiation Agent
<b>Collaborating</b>	Negotiation Agent, Buy/Sell DB Agent, Banker Agent, Buyer Agent, and Seller Agent, Administrator Agent
<b>Data Requirements</b>	
<b>Data Required</b>	Negotiation quotation (Time, place and price offer)

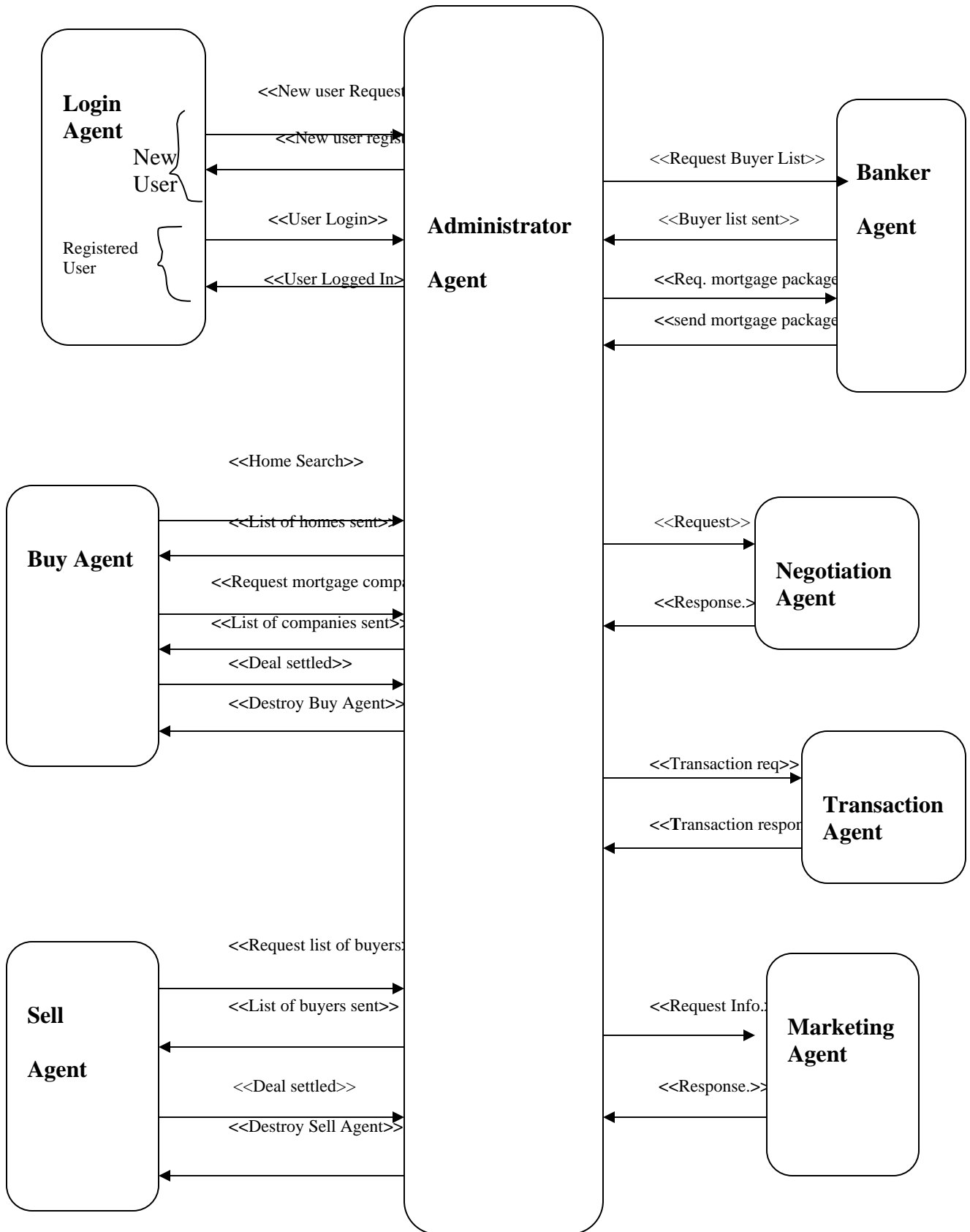
**Marketing Agent:**



**Use case definition:**

<b>Brief description</b>	Administrator Agent uses this use case to update marketing signals in real estate industry
<b>Pre Condition</b>	Electronic news paper should work with electronic realtor
<b>Process steps</b>	
1	Administrator Agent requests to Marketing Agent to update real estate information
2	Marketing Agent collects information from Administrator Agent
3	This updates news paper and magazines dynamically on every transaction
<b>Exceptions</b>	
1	Error message is generated if database is not accessible
<b>Relationship</b>	
<b>Initiating</b>	Administrator Agent
<b>Collaborating</b>	Administrator Agent

**Administrator Agent:**

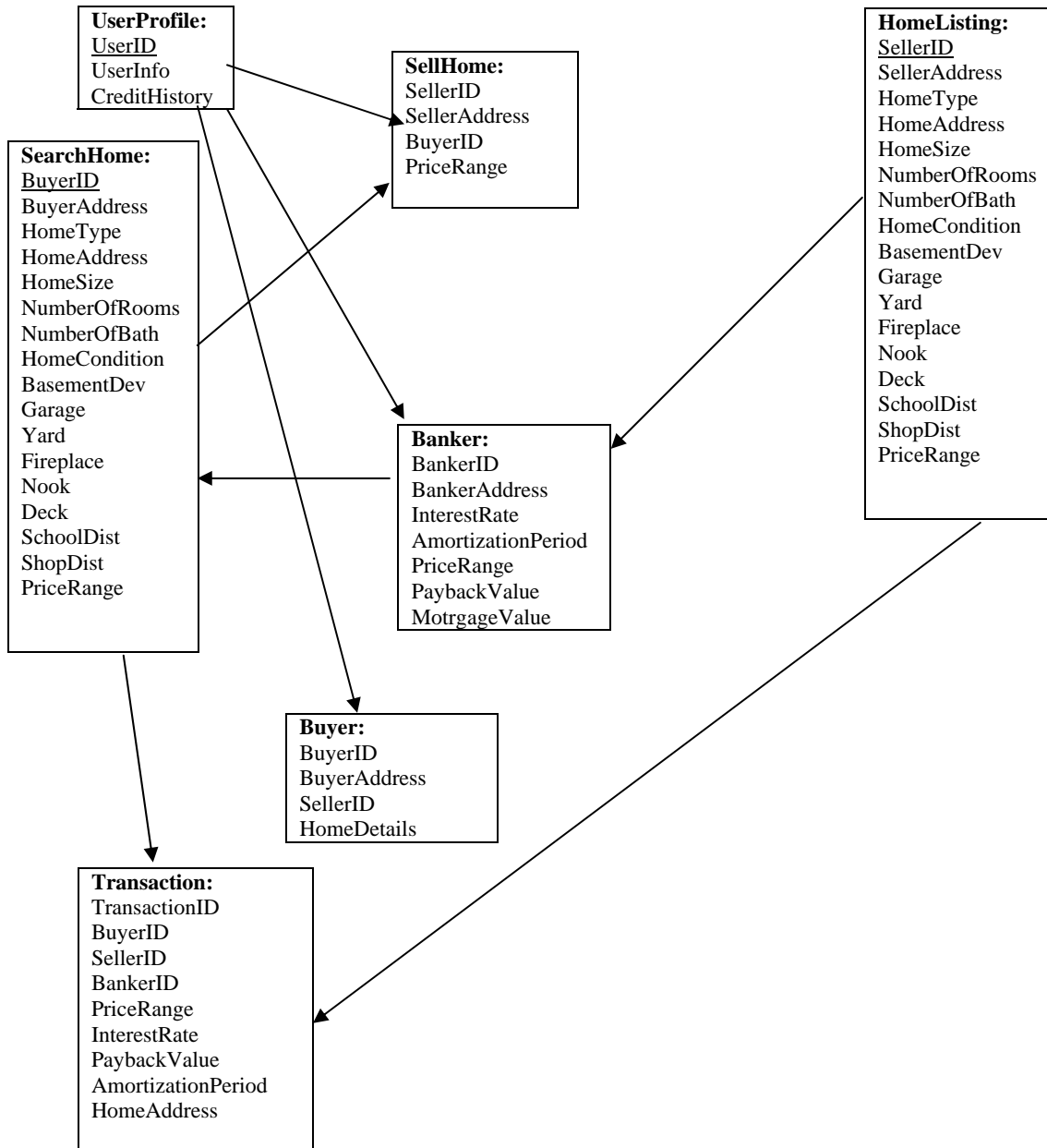


**Use case definition:**

<b>Brief description</b>	All agents use this use case to coordinate activities.
<b>Process steps</b>	
	Steps are given in all other use case definitions above
<b>Relationship</b>	
<b>Initiating</b>	All other agents in the system
<b>Collaborating</b>	All other agents in the system

## 6. Data Specification

### 6.1 Electronic Realtor Data Diagram:



## 6.2 Typical Data Definition:

### Home Listing:

Information for home listing table in this system is as follows:

<b>Field</b>	<b>Description</b>	<b>Type</b>
<b>SellerID</b>	<b>Unique ID</b>	Varchar[50]
SellerAddress	Seller Address	Varchar[50]
HomeType	Home Type	Varchar[50]
HomeAddress	Home address	Varchar[50]
HomeLocation	Home locality	Float
HomeSize	Home Size in sq. ft.	Number
NumberOfRooms	Total number of rooms	Number
NumberOfBath	Total number of bathrooms	Varchar[50]
HomeCondition	Condition of the home	Varchar[50]
BasementDev	Basement finished or unfinished	Number
Garage	Two/single car garage	Float
Yard	Size of backyard	Varchar[50]
Fireplace	Fireplace availability	Varchar[10]
Nook	Nook availability	Float
Deck	Deck availability	Float
SchoolDist	School distance from home	Number
ShopDist	Shop distance from home	
PriceRange	Price of the home	

**Search Home:**

Information for search home table in this system is as follows:

<b>Field</b>	<b>Description</b>	<b>Type</b>
<b>BuyerID</b>	<b>Unique ID</b>	Varchar[50]
BuyerAddress	Buyer Address	Varchar[50]
HomeType	Home Type	Varchar[50]
HomeAddress	Home address	Varchar[50]
HomeLocation	Home locality	Float
HomeSize	Home Size in sq. ft.	Number
NumberOfRooms	Total number of rooms	Number
NumberOfBath	Total number of bathrooms	Varchar[50]
HomeCondition	Condition of the home	Varchar[50]
BasementDev	Basement finished or unfinished	Number
Garage	Two/single car garage	Float
Yard	Size of backyard	Varchar[50]
Fireplace	Fireplace availability	Varchar[10]
Nook	Nook availability	Varchar[10]
Deck	Deck availability	Float
SchoolDist	School distance from home	Float
ShopDist	Shop distance from home	Number
PriceRange	Price of the home	

**Sell Home:**

Information for Sell home table in this system is as follows:

<b>Field</b>	<b>Description</b>	<b>Type</b>
<b>SellerID</b>	<b>Unique ID</b>	Varchar[50]
SellerAddress	Seller Address	Varchar[50]
BuyerID	Unique ID	Varchar[50]
PriceRange	Price of the home	Number

**Buy Home:**

Information for Buy home table in this system is as follows:

<b>Field</b>	<b>Description</b>	<b>Type</b>
<b>BuyerID</b>	<b>Unique ID</b>	Varchar[50]
BuyerAddress	Buyer Address	Varchar[50]
SellerID	Unique ID	Varchar[50]
HomeDetails	Details on home	Varchar[100]

**User Profile:**

Information for User Profile table in this system is as follows:

<b>Field</b>	<b>Description</b>	<b>Type</b>
<b>UserID</b>	<b>Unique ID</b>	Varchar[50]
UserInfo	User Information	Varchar[100]
CreditHistory	Unique ID	Varchar[100]

**Transaction:**

Information for Transaction table in this system is as follows:

<b>Field</b>	<b>Description</b>	<b>Type</b>
<b>TransactionID</b>	<b>Unique ID</b>	Varchar[50]
<b>BuyerID</b>	Buyer ID	Varchar[50]
BankerID	BankerID	Varchar[50]
SellerID	Seller ID	Varchar[100]
PriceRange	Price of home	Number
InterestRate	Interest Rate on mortgage	Float
AmortizationPeriod	Amortization Period	Number
PaybackValue	Payback value	Number

**Banker:**

Information for Banker table in this system is as follows:

Field	Description	Type
<b>BankerID</b>	<b>Unique ID</b>	Varchar[50]
<b>BankerAddress</b>	Buyer ID	Varchar[50]
PriceRange	Price of home	Number
InterestRate	Interest Rate on mortgage	Float
AmortizationPeriod	Amortization Period	Number
PaybackValue	Payback value	Number
MortgagePackage	Mortgage Package	Varchar[100]

## 7. Inter Agent Messages for the Electronic Realtor System

As described in above documents SOAP will be used as a communication protocol between agents and web services. SOAP uses XML as a communication language. Input Output parameters are as follows in XML format:

***List Home:*****Input Parameters:**

Parameter	Description
<pre> &lt;home&gt;   &lt;homeLocation&gt;String &lt;/homeLocation&gt;   &lt;homeAddress&gt;String &lt;/homeAddress&gt;   &lt;homeSize&gt;Float&lt;/homeSize&gt;    &lt;NumberOfRooms&gt;Number&lt;/NumberOfRooms&gt;   &lt;NumberOfBath&gt; Number &lt;/NumberOfBath&gt;   &lt;HomeCondition&gt;String &lt;/HomeCondition&gt;   &lt;BasementDev&gt;String &lt;/BasementDev&gt;   &lt;Garage&gt;String&lt;/Garage&gt;   &lt; Yard &gt;String&lt;/ Yard &gt;   &lt; Fireplace &gt;String&lt;/ Fireplace &gt;   &lt; Nook &gt;String&lt;/ Nook &gt;   &lt; Deck &gt;String&lt;/ Deck &gt;   &lt; SchoolDist &gt;String&lt;/ SchoolDist &gt;   &lt; ShopDist &gt;String&lt;/ ShopDist &gt;   &lt;PriceRange&gt;Number &lt;/PriceRange&gt; &lt;/home&gt; </pre>	Information related home

**Output Parameters:**

Parameter	Description
<homeBuyer> <BuyerID>String </ BuyerID > <PriceRange>Number </PriceRange> <homeSearch> String </homeSearch> </ homeBuyer >	Information related candidate home buyers

**Search Home:****Input Parameters:**

Parameter	Description
<searchHome> <homeLocation>String </homeLocation> <homeAddress>String </homeAddress> <homeSize>Float</homeSize>  <NumberOfRooms>Number</NumberOfRooms> <NumberOfBath> Number </NumberOfBath> <HomeCondition>String </HomeCondition> <BasementDev>String </BasementDev> <Garage>String</Garage> < Yard >String</ Yard > < Fireplace >String</ Fireplace > < Nook >String</ Nook > < Deck >String</ Deck > < SchoolDist >String</ SchoolDist > < ShopDist >String</ ShopDist > <PriceRange>Number </PriceRange> </ searchHome >	Information related home

**Output Parameters:**

Parameter	Description
<homeBuyer> <BuyerID>String </ BuyerID > <BuyerAddress>String </ BuyerAddress > <PriceRange>Number </PriceRange> <homeSearch> String </homeSearch> </ homeBuyer >	Information related candidate home buyers

**Buy Home:****Input Parameters:**

Parameter	Description
<BuyHome> <buyerID>String </ buyerID > <buyerAddress>String </ buyerAddress > <sellerID> String </ sellerID > <homeDetails> String </ homeDetails > </ BuyHome >	Information related buying home

**Output Parameters:**

Parameter	Description
<homeBuyer> <BuyerID>String </ BuyerID > <BuyerAddress>String </ BuyerAddress > <PriceRange>Number </PriceRange> <homeSearch> String </homeSearch> </ homeBuyer >	Information related candidate home buyers

**Acquire Mortgage:****Input Parameters:**

Parameter	Description
<homeBuyer> <buyerID>String </ buyerID > <buyerAddress>String </ buyerAddress > <buyerIncome> String </ buyerIncome > <homePrice> String </ homePrice > </ homeBuyer >	Acquire mortgage from banker

**Output Parameters:**

Parameter	Description
<banker> < banker ID>String </ banker ID > <mortgageConditions>String <mortgageValue>Number </ mortgageValue > <interestRate> Float </ interestRate > <amortizationPeriod> Float </ amortizationPeriod > </mortgageConditions > </ banker >	Information related mortgage package and mortgage company

## **Conclusions:**

While working on this project we had an opportunity to gain experience in applying agent-based software design concepts learnt in tutorial session. Although overall design architecture is done, more detailed and left out components are require be completing and implementing. A practical implementation is worth to carry on.

Since electronic realtor is going to be one of the consumer demands for sophistication system based on electronic business, the potential of agent-based system is very promising.