Auction House Design

Design Doc for the Auction House assignment

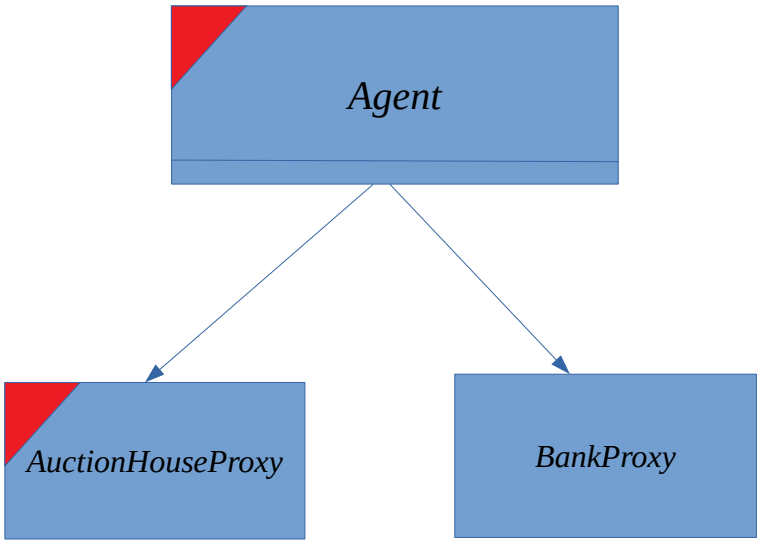
Group Members:

Farhang Rouhi

Alexander Booher

Charles Habermehl

Agent



Agent Components:

Agent: This part has all of the logic and user interacts with this part using the terminal. The commands are:

1. log out: it logs out and terminates the program peacefully, if no bids are waiting.
2. get auction houses: it gets and prints the auction houses.
3. get auctions from "auction house number": this one gets auctions from the selected auction house.
 Note that the number is the number printed when calling get auction houses.
4. bid on "itemNum" for "amount" in "auctionHouse number": This one allows the user to bid on a certain item.
 Note that the itemNum will be printed as the index number or id. The amount is the amount of bid. and the
 auction house number is printed when we print auction houses.

AuctionHouseProxy: This parts talks to the auction house and it also waits for notifications coming from the auction house. It provides functions for easy access of the rest of the program.

BankProxy: This one is also a proxy that talks to the Bank and provides functions. Note that this doesn’t have any threads.

Functions

main: This is my main function which has all of the logic for my agent. It simply waits for user input. User can use the following commands: log out: it logs out and terminates the program. get auction houses: it gets and prints the auction houses. get auctions from "auction house number": this one gets auctions from the selected auction house. Note that the number is the number printed when calling get auction houses. bid on "itemId" for "amount" in "auctionHouse number": This one allows the user to bid on a certain item. Note that the itemId will be printed via the previous command. The amount is the amount of bid. and the auction house number is printed when we print auction houses.

Parameters: name amount BankIp BankPort

isTerminated: returns the termination status.

Terminate: terminates the connection and all dependencies.

Run: This thread run function listens for messages from the auction house. If the message is a notification it puts it in notificationQueue, otherwise the message must be a return value. So, it puts it in the queue.

TakeNotification: This function takes notifications for a certain item. If not related to that item, it puts it back.

Communicate: This function sends messages to the auction house and then receives the return value, by getting it from the queue.

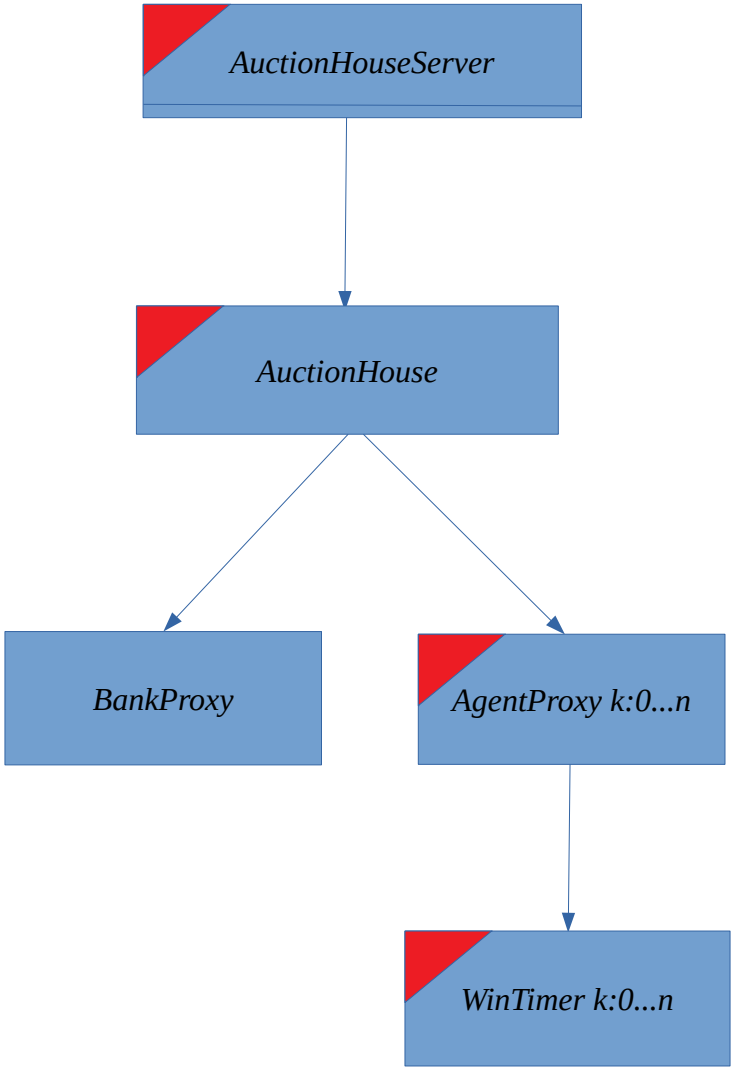
Bid: bids on an item.

GetItems: This functions gets and returns the items from the auction house.

Sendreceipt: sends the receipt of transaction.

ReleaseLock, transferMoney, lockBalance, createAccount, closeAccount and getAuctionHouses are BankProxy functions that have obvious functionalities.

Auction House



Auction house

AuctionHouseServer: This one is the start point of this package. It will wait for new clients and create a proxy for each of them.

AuctionHouse: This is the main class and does all of the book keepings.

BankProxy: This one connects to the bank and provides required functions.

AgentProxy: This one connects to the agent and provides connection to agent.

WinTimer: This waits for 30 seconds on each bid and if not overtaken, that bid will win.

Functions:

Functions of bankproxy are exactly like agents bank proxy. Agent proxy provides all the functionalities in auction house proxy of agent, so the functions are very simmilar.

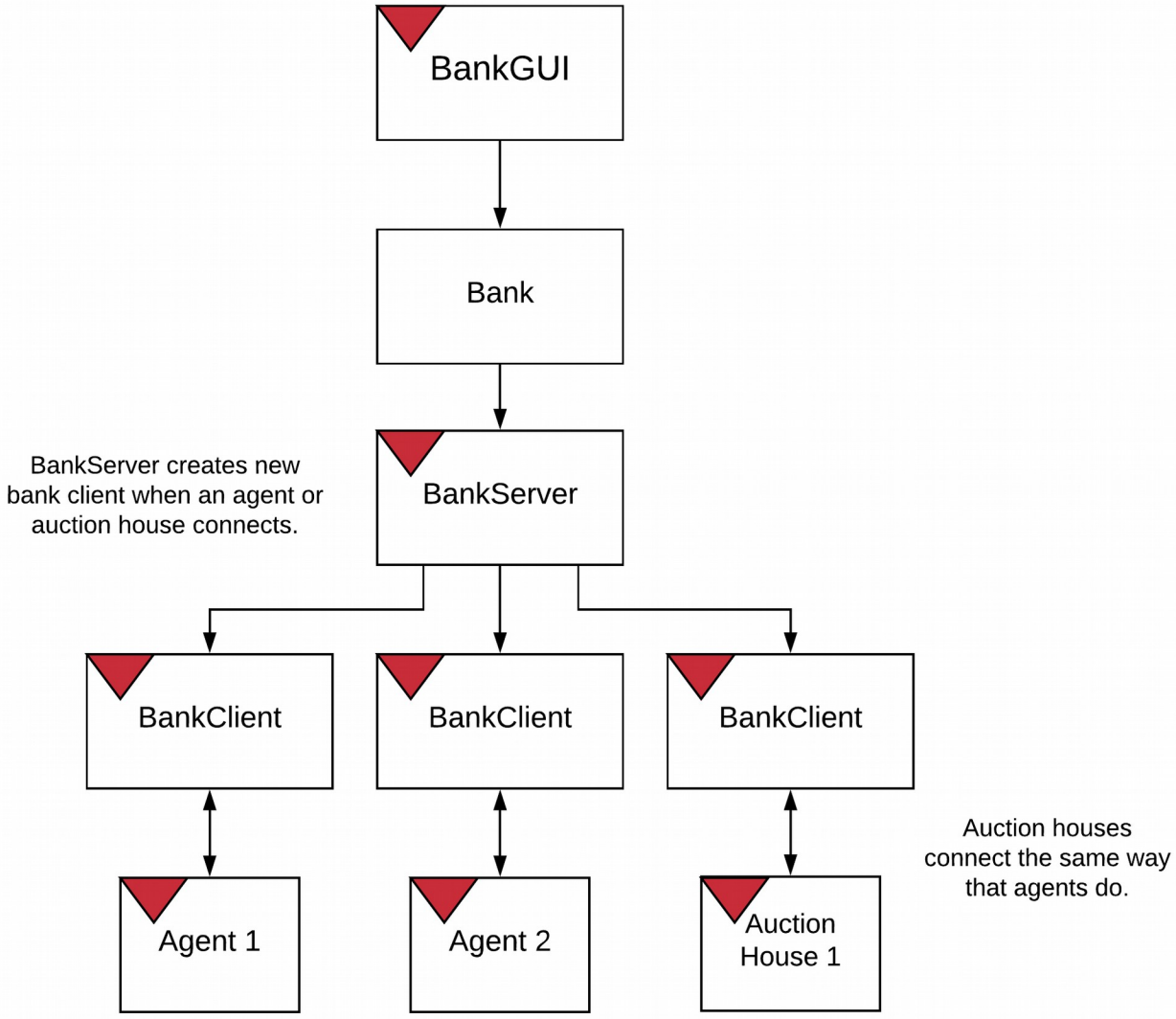
Main: This is the server and waits for agents to come an connect. Parameters: name portNumber BankIp BankPort [optional: local ip]

isBidValid: This will determine if the bid is valid. If it is valid, it will process it buy making a winTimer.

ItemSold and updateReceiptNum are only for book keeping.

GetCurrentBidAmount, getItems and terminate have obvious usages. To learn more, go to the javadocs.

Bank Structure



Bank Functions:

void openNewAccount(String name, double initialDeposit) –Opens a new bank account with the given name

int getAccountNumber() – returns account number

int getBidKey() – returns bid key

Account getAccount(int accountNumber) – returns account by number

void setAccountHold(int bidkey, double bid) – locks funds in the given account by the bid amount.

double getBalance(int accountNumber) – returns the balance of the given account

boolean hasEnoughFunds(int key, double bid) – returns whether the given account has enough funds to make the bid.

boolean isValidNumber(int accountNumber) – returns true if the given account number is valid.

String getAccountDetails(int accountNumber) – returns the account details of the given account number in string form.

String getAuctionAccountDetails(int accountNumber) – returns the account details of the given auction account.

String getAuctionString() – returns a string containing all the available auction houses.

void moveMoney(int accountA, int accountB, double amount) – moves money from one account to the other.

Design structure of Packages: