Auction House

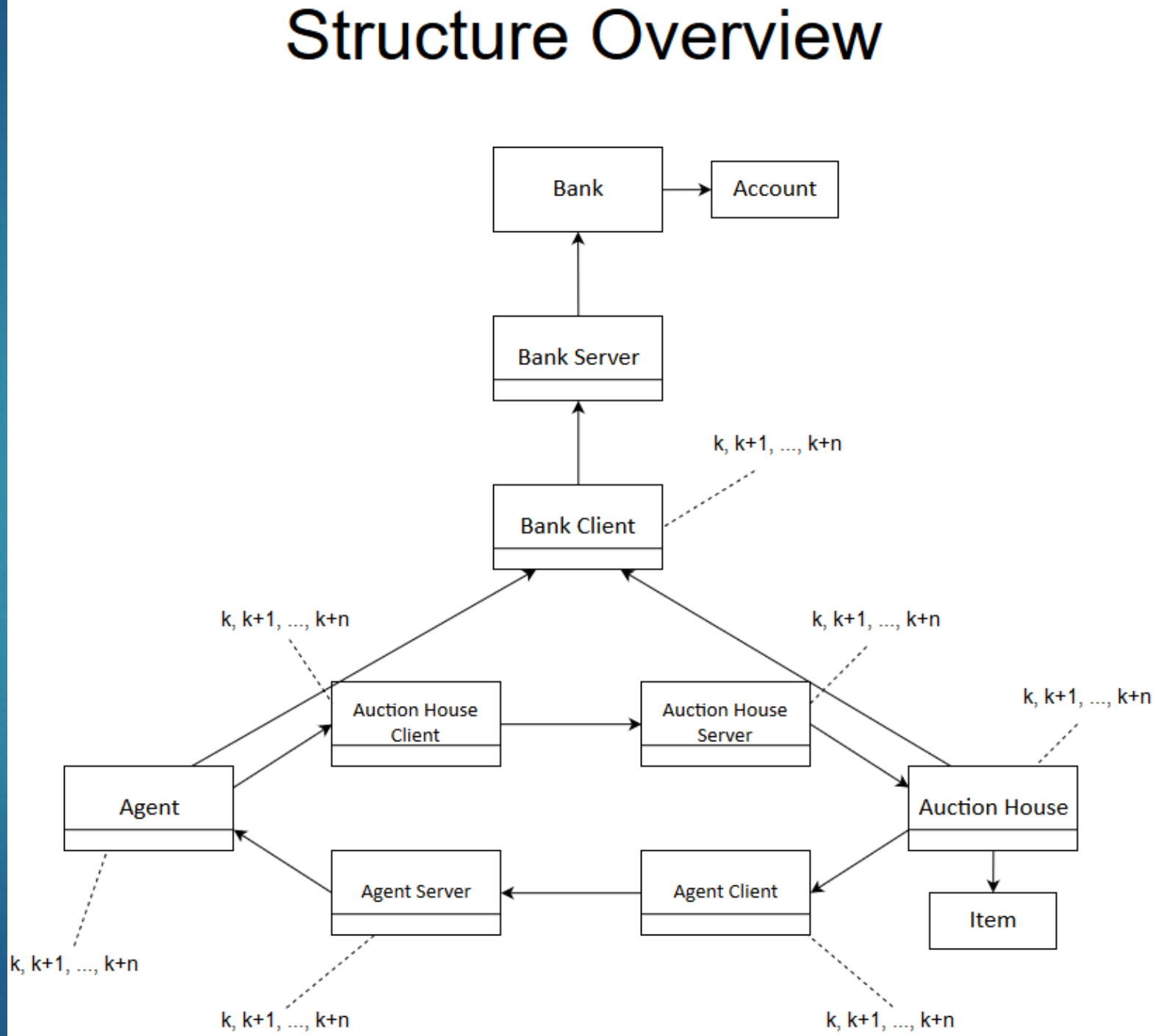
Final Project

CREATED BY: NATHAN SCHAEFER, CLARISSA GARCIA, & ANAS GUABA

CS 351 DECEMBER 5TH, 2018

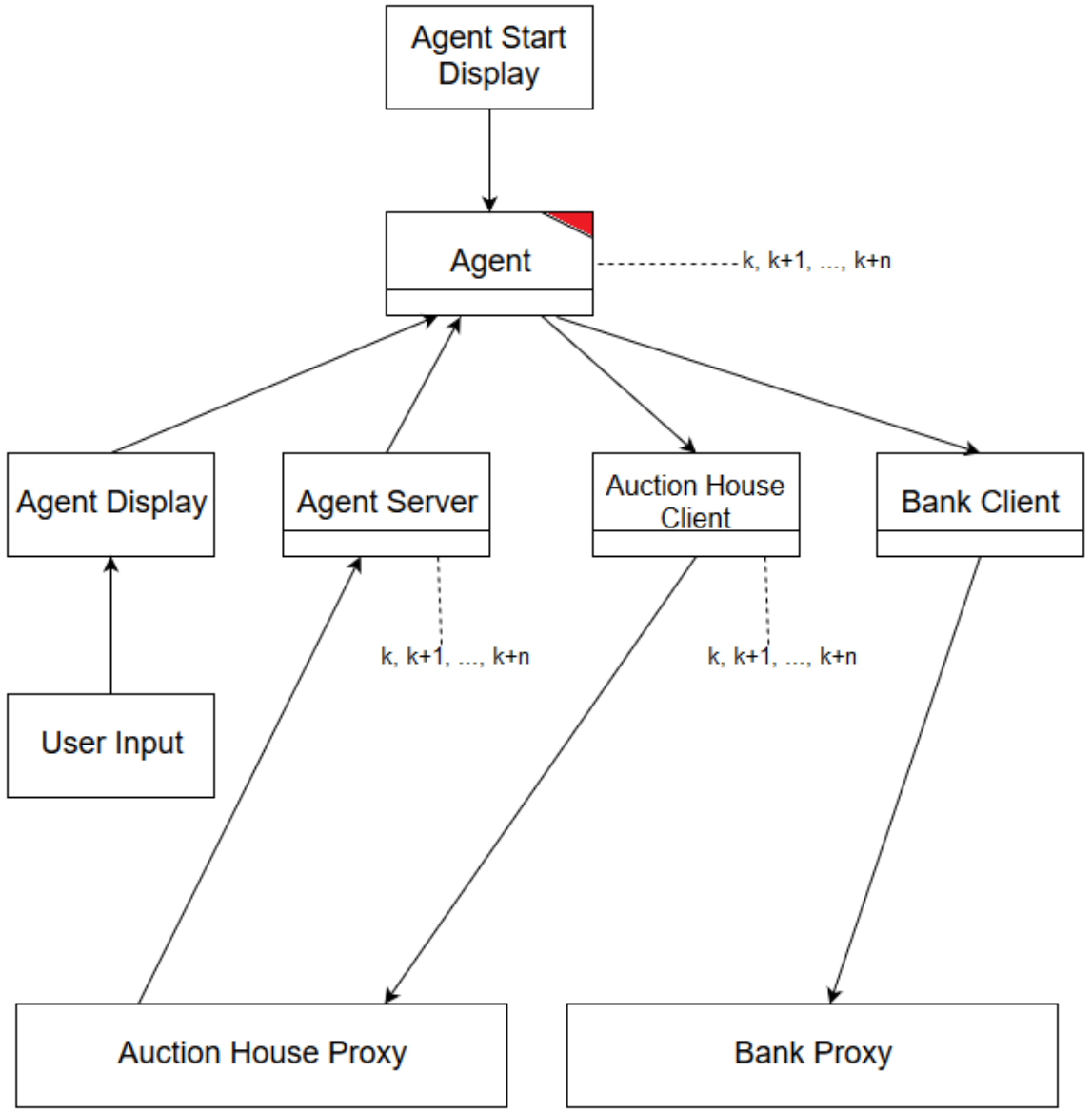
PROFESSORS CHENOWETH & ROMAN

Projects Diagrams - (See Designs PDF for better images)



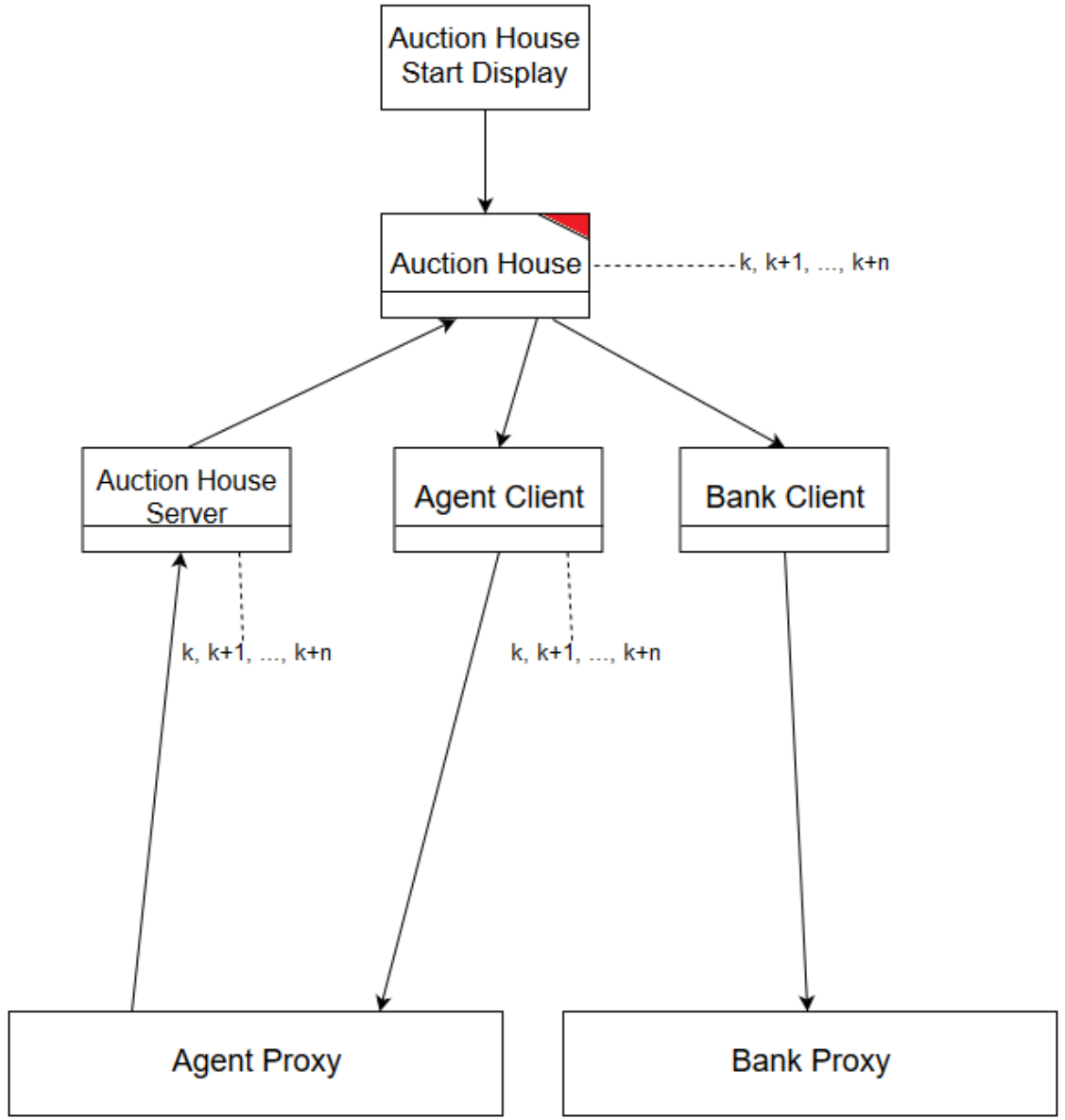
Agent Stucture

(Agent Proxy)



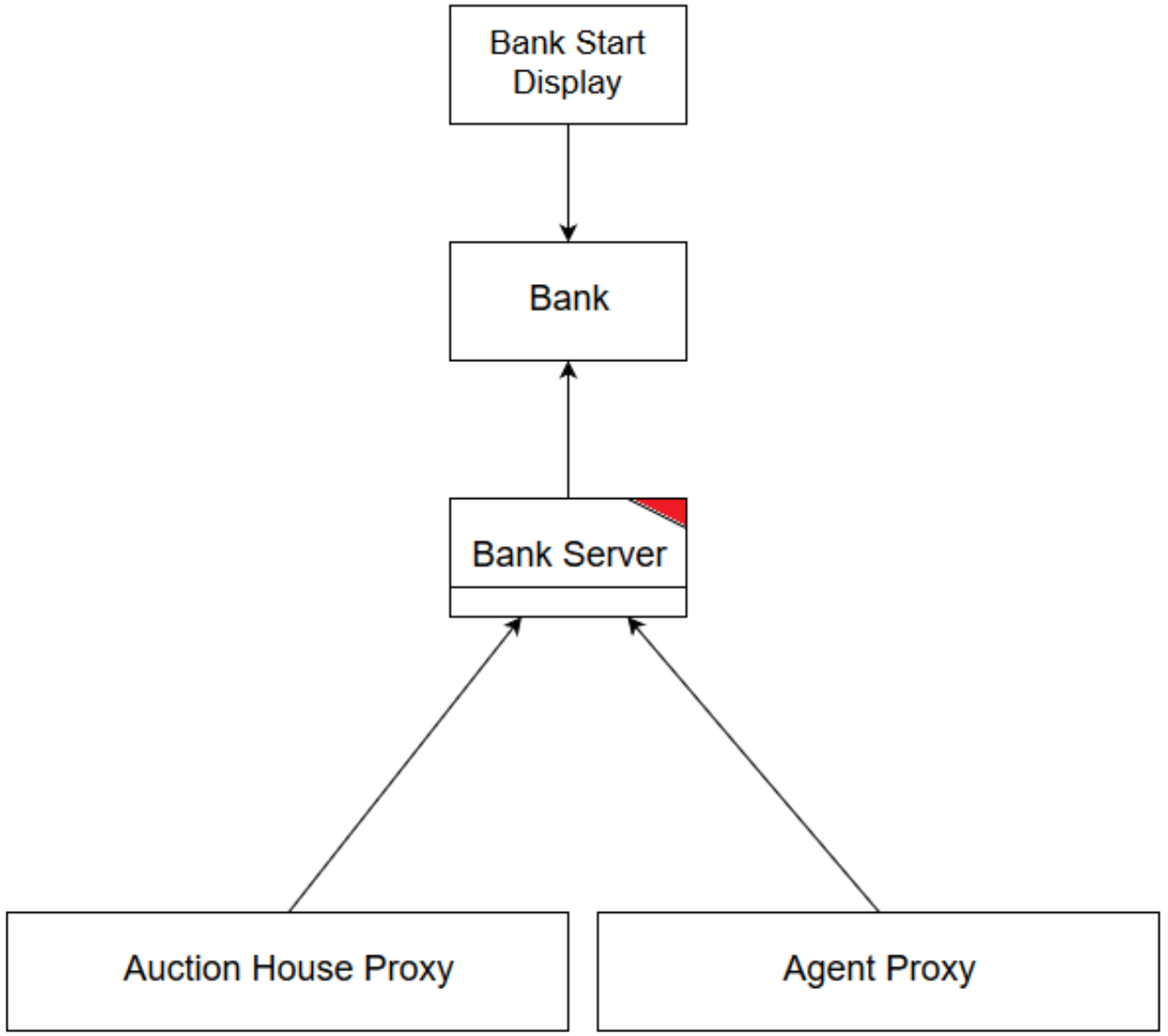
Auction House Stucture

(Auction House Proxy)



Bank Stucture

(Bank Proxy)



Agent Methods

- Agent(int, int, LinkedList<Strings>) – creates agent user
- createName() – gives user a random name
- setBiddingKey(int) – sets agents key from bank
- setAgentDisplayValues(int, double) – sets agents display values
- changeBalance(double) – sets users balance
- sendBid(String, double) – sends Auction House the amount of their bid
- setCurrentAucitonHouse(int) – sets the users current auction house that they selected in gui
- printDetermination(Command, Item) – Used for giving agent notifications
- createItemList(LinkedList<Item> - store a temporary item list retrieved from auction house
- refreshItems() – resets the temporary item list and makes a new list
- refreshTimes() - updates times in gui
- createHouseList() – creates a list of auction houses for the user to select from
- getHouseList(LinkedList<int> HashMap<int, int>) used to create clients for the agent to interact with auction houses
- setTimeOffSet(Long) – gets the time offset from user from auction house server (gui purposes)
- getTimeOffSet()
- closeAccount() – closes agent's account
- Sound(String) – user feedback sounds

Auction House Methods

- AuctionHouse(display reference, int, int, LinkedList<String>, LinkedList<String>) – creates auction house with given ports and uses lists to create random item names.
- setKey(int) – sets the auction houses key on items
- bidSuccessfulCheck() – checks if any items bid has ended
- sendBid(int, string, double, double) – receives agents bid amount and checks if they can do the bid
- createItems(int) – creates amount of items base on int received.
- getItemList() – returns the items list back to agent when requested
- setHasFunds(Boolean) – checks if the user has funds, response received from bank
- setBalance(double) – sets the auctionHouses balance in the gui
- setPortNumber(int)
- stopAuctionHouse()
- removeAgent(int) used when agent closes account, so auction house can remove the user from active users
- startAuctionHouseClient(String) used to commuinicate back to agents

Item Methods

- Item(String, String, double, double) – creates the item and store auction house info within.
- getAuctionHouseID()
- getItemID()
- getDescription
- getMinimumBidAmount()
- getCurrentBidAmount()
- getBidTimeRemaining()
- setBidAmount()
- setSecretBidderKey(int) – agents bidder key
- getSecretBidderKey()
- startBidTime()
- setAuctionActive(Boolean)
- getAuctionActive()
- setAuctionHouseSecretKey(int)
- getAuctionHouseSecretKey()
- toString() – override and return string info on item

Bank Methods

- Bank(reference of bank, int) – creates a bank with a given port number
- createAccount(String, double) – creates an account with a unique secret key and account number
- getBalance()
- abilityToBuy(int double) – checks if given secret key linked to account have funds to buy an item
- lockBalance(int, double) locks a users balance when in an active bid
- unLockBalance(int, double) – unlocks users amount of funds when passed in a bid
- deposit(int, int, double) – takes funds from one account and puts in another when a user wins a bid and gives the balance to the auction house.
- closeAccount(int) – removes the account from the list of active accounts

Account Methods

- Account(String, Double) – creates a new account with the give string name and intitial balance,
- getAccountID()
- setAccountID(int)
- setSecretKey()
- generateSecretKeyt()

All Clients

- All clients that are being initialized will first send out a string to the server they're connecting to and give info about client
- Client info will include the port number it's connecting to, where it's connecting from, and the client type.
- This is done so that server like Auction House can initialize a client back to agent.
- After the initial message and client message, it gets thrown into a loop that waits for messages that send commands to the respective server

All Servers

- Like agent clients, they wait for an initial message from the clients.
- When the server does receive a message from the client, it will first create a new thread of that server to handle that client connection.
- After the server is ready to accept the clients message, it translate the info given.
- Depending on the server type component it will start a client back to the server (Like the agent to auction house relationship)
- Bank never creates a client back to anyone, as bank's server doesn't ever need to active send alerts to any component like auction house to agent.