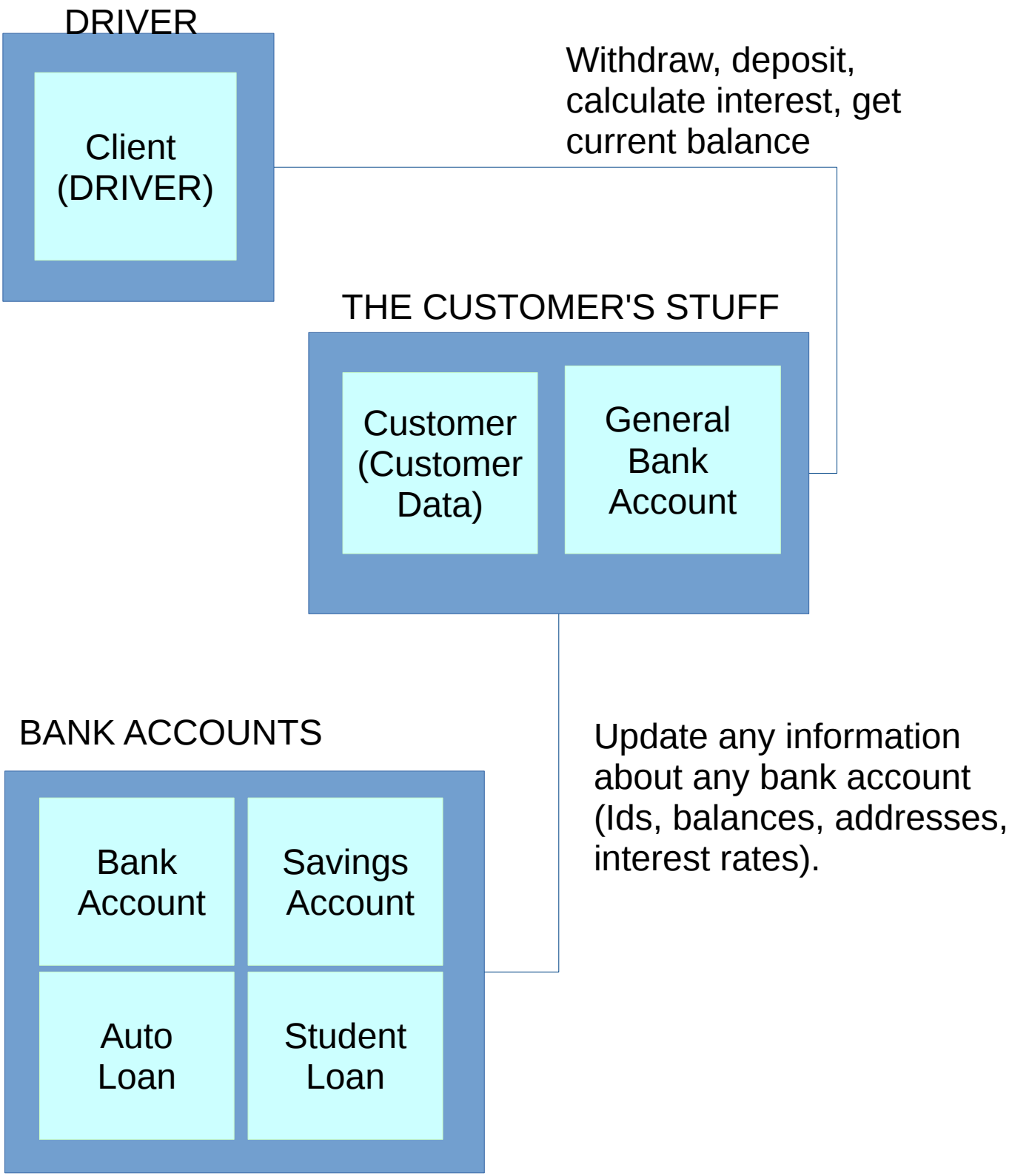
Analysis:
Our program is centered around the Bank Account class. We must create a Client class which works as the driver. There are four types of bank accounts, which should extend the Bank Account class: A checkings account, and three accounts that should extend a Savings Account class: a primary savings, student loan, and auto loan account. We must handle input and output summary messages about what has happened during the transaction(s). We also need a Customer class to keep track of customers. Accounts have unique properties and only the balance changes.

Design:
System IPO diagram

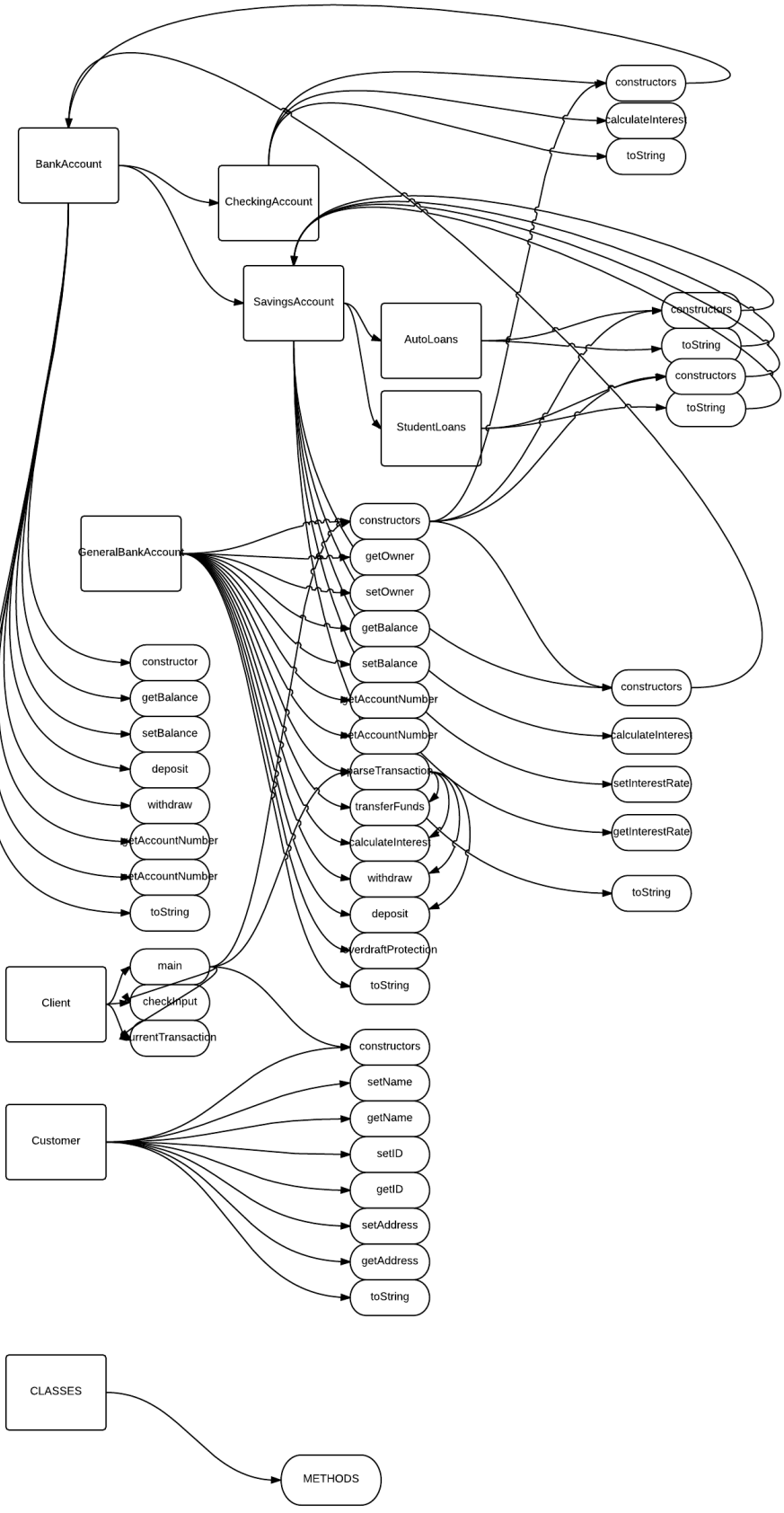
INPUT	PROCESS	OUTPUT
A string command in the form **ID# transaction-type [amount]account-type [account-type2]**	The string is checked for correctness by the Client Class The string is translated into a transaction type by the Client Class The requested transaction is executed by the General Bank Account Class The user is given the option to make another transaction or quit (restart or end)	If the string is incorrect, an error message is displayed If the transaction completes successfully the updated information is displayed and another transaction is possible If the client terminates, a goodbye message is displayed and the current state of all customer accounts is printed to the console

Functional block diagram

Everything about this is extremely ambiguous. I've never seen a functional block diagram before in my life, everything I've read has led me to believe that it's meant only for logic design, and piazza answers aren't helpful. Here is what I think you want:



And here is satan's spawn, a diagram that matches every method with every method it calls:



Algorithm description of driver:
1.) Initialize bank accounts, add ten possible customers with made up addresses and names and IDs0--9
2.) Show a text input that waits for input
2b.) If the user selects cancel, exit (GO TO 5)
2c.) Make sure the input isn't empty (if it is, print an error message)
3.) If the input is valid and the transaction can happen, execute the transaction-type (otherwise print an error message and GO TO 2)
4.) if the user selects quit GO TO 5 otherwise GO TO 2
5.) print out each user's info and bank accounts and display goodbye message