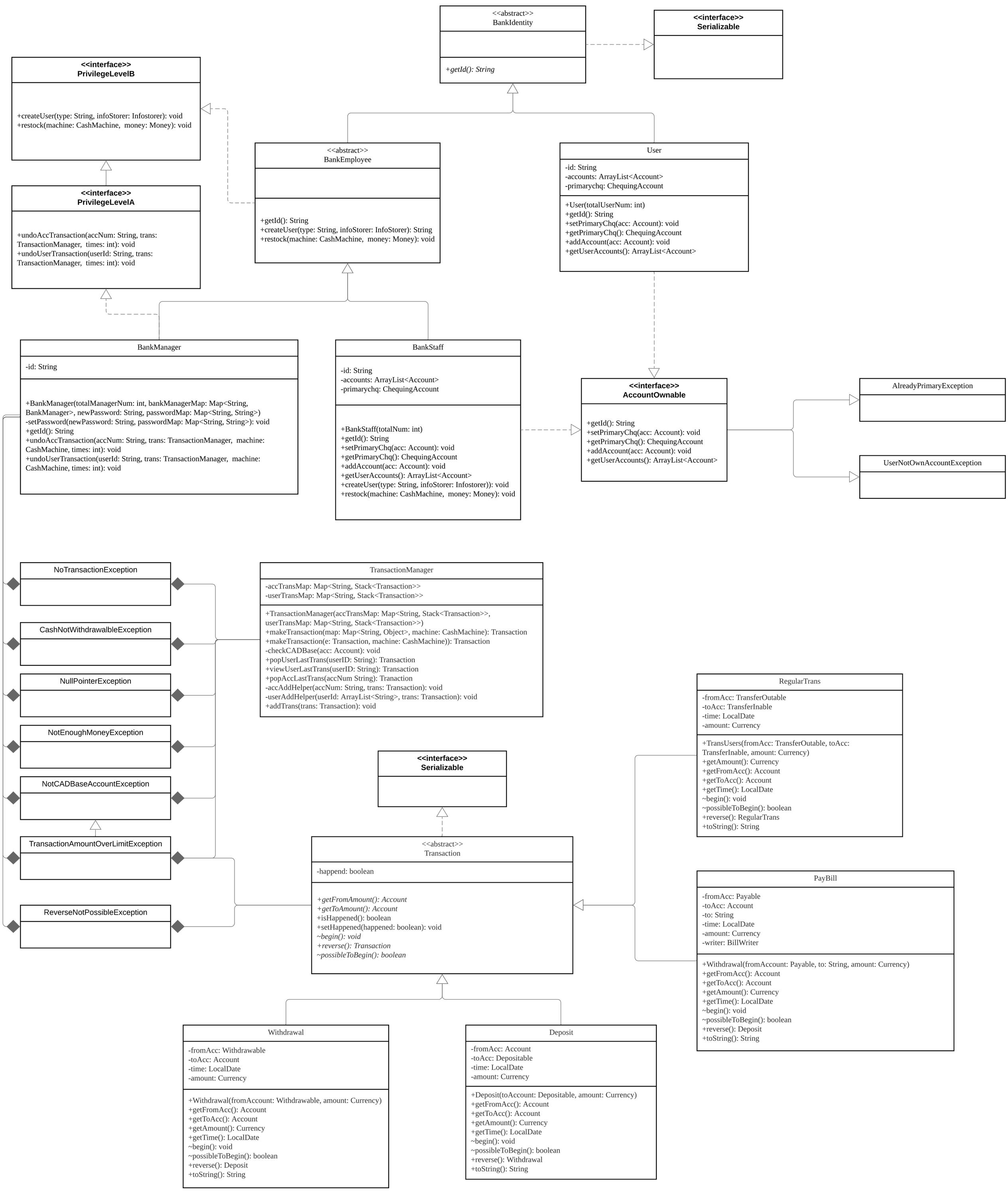
<<abstract>>
BankIdentity
+getId(): String
<<interface>>
Serializable
<<interface>>
PrivilegeLevelB
+createUser(type: String, infoStorer: Infostorer): void
+restock(machine: CashMachine, money: Money): void
<<interface>>
PrivilegeLevelA
+undoAccTransaction(accNum: String, trans: TransactionManager, times: int): void
+undoUserTransaction(userId: String, trans: TransactionManager, times: int): void
<<abstract>>
BankEmployee
+getId(): String
+createUser(type: String, infoStorer: InfoStorer): String
+restock(machine: CashMachine, money: Money): void
User
-id: String
-accounts: ArrayList<Account>
-primarychq: ChequingAccount
+User(totalUserNum: int)
+getId(): String
+setPrimaryChq(acc: Account): void
+getPrimaryChq(): ChequingAccount
+addAccount(acc: Account): void
+getUserAccounts(): ArrayList<Account>
BankManager
-id: String
+BankManager(totalManagerNum: int, bankManagerMap: Map<String, BankManager>, newPassword: String, passwordMap: Map<String, String>)
-setPassword(newPassword: String, passwordMap: Map<String, String>): void
+getId(): String
+undoAccTransaction(accNum: String, trans: TransactionManager, machine: CashMachine, times: int): void
+undoUserTransaction(userId: String, trans: TransactionManager, machine: CashMachine, times: int): void
BankStaff
-id: String
-accounts: ArrayList<Account>
-primarychq: ChequingAccount
+BankStaff(totalNum: int)
+getId(): String
+setPrimaryChq(acc: Account): void
+getPrimaryChq(): ChequingAccount
+addAccount(acc: Account): void
+getUserAccounts(): ArrayList<Account>
+createUser(type: String, infoStorer: Infostorer)): void
+restock(machine: CashMachine, money: Money): void
<<interface>>
AccountOwnable
+getId(): String
+setPrimaryChq(acc: Account): void
+getPrimaryChq(): ChequingAccount
+addAccount(acc: Account): void
+getUserAccounts(): ArrayList<Account>
AlreadyPrimaryException
UserNotOwnAccountException
NoTransactionException
CashNotWithdrawalbleException
NullPointerException
NotEnoughMoneyException
NotCADBaseAccountException
TransactionAmountOverLimitException
ReverseNotPossibleException
TransactionManager
-accTransMap: Map<String, Stack<Transaction>>
-userTransMap: Map<String, Stack<Transaction>>
+TransactionManager(accTransMap: Map<String, Stack<Transaction>>, userTransMap: Map<String, Stack<Transaction>>)
+makeTransaction(map: Map<String, Object>, machine: CashMachine): Transaction
+makeTransaction(e: Transaction, machine: CashMachine)): Transaction
-checkCADBase(acc: Account): void
+popUserLastTrans(userID: String): Transaction
+viewUserLastTrans(userID: String): Transaction
+popAccLastTrans(accNum String): Tranaction
-accAddHelper(accNum: String, trans: Transaction): void
-userAddHelper(userId: ArrayList<String>, trans: Transaction): void
+addTrans(trans: Transaction): void
RegularTrans
-fromAcc: TransferOutable
-toAcc: TransferInable
-time: LocalDate
-amount: Currency
+TransUsers(fromAcc: TransferOutable, toAcc: TransferInable, amount: Currency)
+getAmount(): Currency
+getFromAcc(): Account
+getToAcc(): Account
+getTime(): LocalDate
~begin(): void
~possibleToBegin(): boolean
+reverse(): RegularTrans
+toString(): String
<<interface>>
Serializable
<<abstract>>
Transaction
-happend: boolean
+getFromAmount(): Account
+getToAmount(): Account
+isHappened(): boolean
+setHappened(happened: boolean): void
~begin(): void
+reverse(): Transaction
~possibleToBegin(): boolean
PayBill
-fromAcc: Payable
-toAcc: Account
-to: String
-time: LocalDate
-amount: Currency
-writer: BillWriter
+Withdrawal(fromAccount: Payable, to: String, amount: Currency)
+getFromAcc(): Account
+getToAcc(): Account
+getAmount(): Currency
+getTime(): LocalDate
~begin(): void
~possibleToBegin(): boolean
+reverse(): Deposit
+toString(): String
Withdrawal
-fromAcc: Withdrawable
-toAcc: Account
-time: LocalDate
-amount: Currency
+Withdrawal(fromAccount: Withdrawable, amount: Currency)
+getFromAcc(): Account
+getToAcc(): Account
+getAmount(): Currency
+getTime(): LocalDate
~begin(): void
~possibleToBegin(): boolean
+reverse(): Deposit
+toString(): String
Deposit
-fromAcc: Account
-toAcc: Depositable
-time: LocalDate
-amount: Currency
+Deposit(toAccount: Depositable, amount: Currency)
+getFromAcc(): Account
+getToAcc(): Account
+getAmount(): Currency
+getTime(): LocalDate
~begin(): void
~possibleToBegin(): boolean
+reverse(): Withdrawal
+toString(): String