8.3 AUTOMATED TELLER MACHINE EXAMPLE

The following problem statement for an automated teller machine (ATM) network shown in Figure 8.3 serves as an example throughout the chapter:
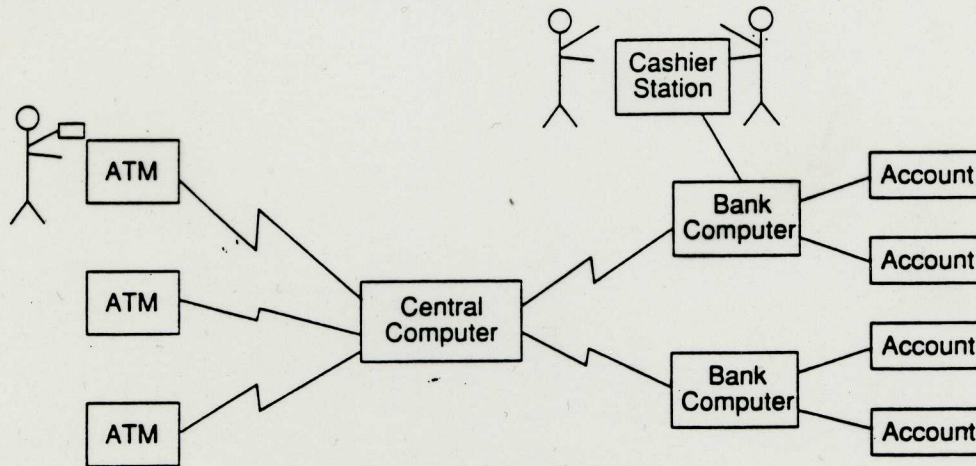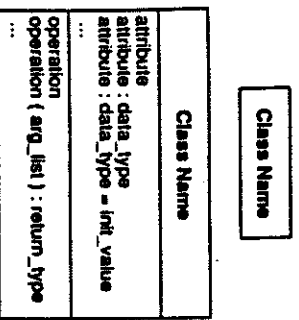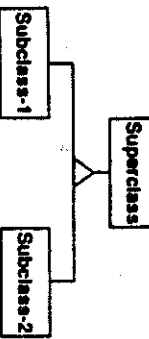

Figure 8.3 ATM network

Design the software to support a computerized banking network including both human cashiers and automatic teller machines (ATMs) to be shared by a consortium of banks. Each bank provides its own computer to maintain its own accounts and process transactions against them. Cashier stations are owned by individual banks and communicate directly with their own bank's computers. Human cashiers enter account and transaction data. Automatic teller machines communicate with a central computer which clears transactions with the appropriate banks. An automatic teller machine accepts a cash card, interacts with the user, communicates with the central system to carry out the transaction, dispenses cash, and prints receipts. The system requires appropriate recordkeeping and security provisions. The system must handle concurrent accesses to the same account correctly. The banks will provide their own software for their own computers: you are to design the software for the ATMs and the network. The cost of the shared system will be apportioned to the banks according to the number of customers with cash cards.
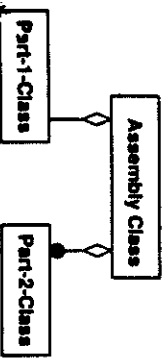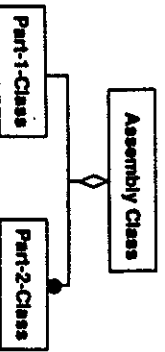
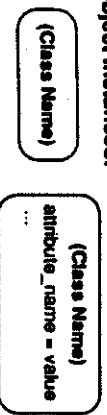# Object Model Notation
Basic Concepts

Class:

Class Name

Class Name
| attribute |
| attribute : data_type |
| attribute : data_type = init_value |
| operation |
| operation (arg_list) : return_type |
| ... |

Generalization (inheritance):

Superclass — Subclass-1, Subclass-2

Aggregation:

Assembly Class — Part-1-Class, Part-2-Class

Aggregation (alternate form):

Assembly Class — Part-1-Class, Part-2-Class

Object Instances:

(Class Name)

(Class Name)
attribute_name = value
...

Association:

Class-1 — Association Name — Class-2
role-1   role-2

Qualified Association:

Class-1 | qualifier — Association Name — Class-2
role-1   role-2
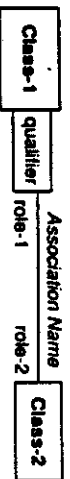
Multiplicity of Associations:

Class — Exactly one
Class — Many (zero or more)
Class — Optional (zero or one)
Class — One or more — 1+
Class — Numerically specified — 1-2, 4

Ordering:

{ordered}
Class

Link Attribute:
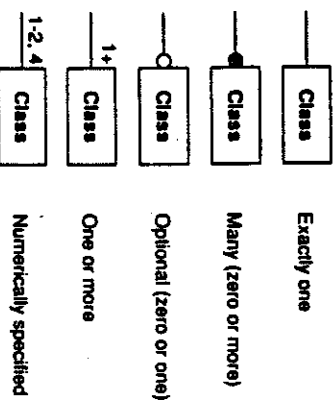
Class-1 — Class-2
link attribute

Ternary Association:

Class-1 — Association Name — Class-2
role-1   role-2
role-3
Class-3
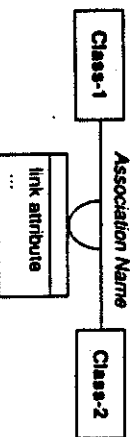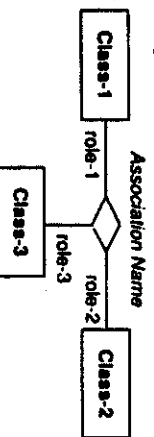
Instantiation Relationship:

(Class Name) ·······> Class Name
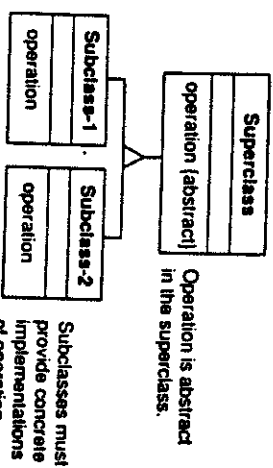
Object Model Notation
Advanced Concepts

Abstract Operation:

Superclass
| operation {abstract} | — Operation is abstract in the superclass.

Subclass-1
| operation |
Subclass-2
| operation |
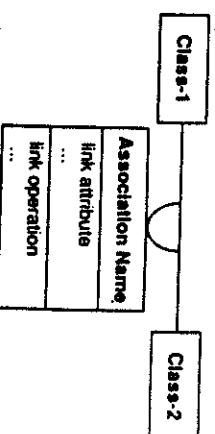
Generalization Properties:

Subclasses must provide concrete implementations of operation.

Superclass — Subclass-1, Subclass-2
Subclasses have overlapping (nondisjoint) membership.

Superclass
discriminator
Subclass-1, Subclass-2
Discriminator is an attribute whose value differentiates between subclasses.

Multiple Inheritance:

Superclass-1, Superclass-2 — Subclass
... More subclasses exist.

Class Attributes and Class Operations:

Class Name
| $attribute |
| $operation |
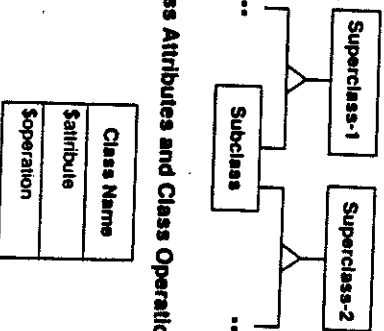
Derived Attribute:

Class Name
| /attribute |

Derived Class:

/Class Name

Propagation of Operations:

Class-1 | operation — operation → Class-2 | operation

Derived Association:

Class-1 —/— Class-2

Constraints on Objects:
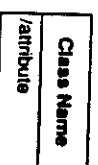
Class-1
| attrib-1 |
| attrib-2 |
{ attrib-1 ≥ 0 }

Constraint between Associations:

Class-1 — A1 {subset}, A2 — Class-2

Association as Class:

Class-1 — Association Name — Class-2
Association Name
| link attribute |
| link operation |
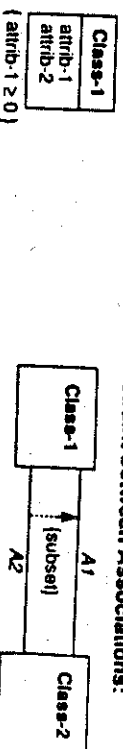
The following steps are performed in constructing an object model:

- Identify objects and classes [8.4.1-8.4.2]
- Prepare a data dictionary [8.4.3]
- Identify associations (including aggregations) between objects [8.4.4-8.4.5]
- Identify attributes of objects and links [8.4.6-8.4.7]
- Organize and simplify object classes using inheritance [8.4.8]
- Verify that access paths exist for likely queries [8.4.9]
- Iterate and refine the model [8.4.10]
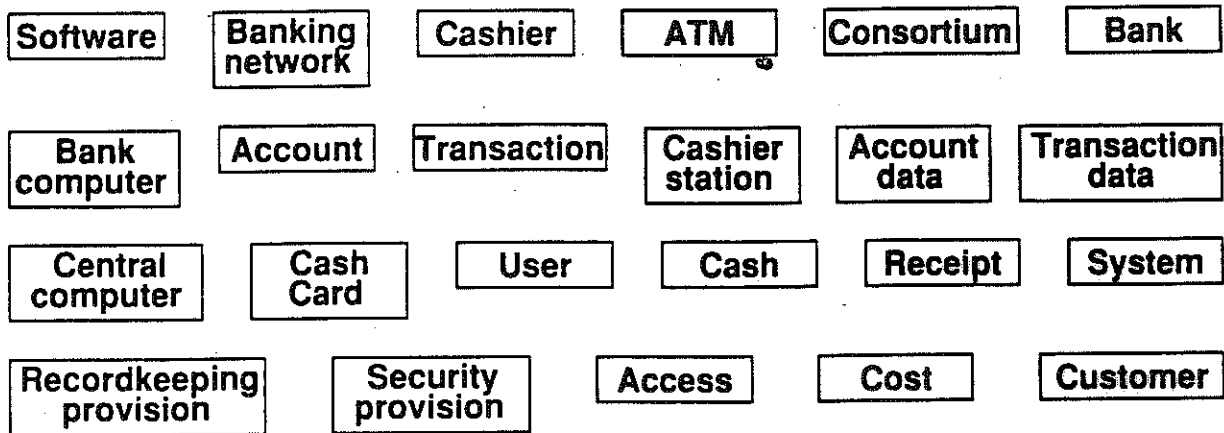- Group classes into modules [8.4.11]

Software	Banking network	Cashier	ATM	Consortium	Bank

Bank computer	Account	Transaction	Cashier station	Account data	Transaction data

Central computer	Cash Card	User	Cash	Receipt	System

Recordkeeping provision	Security provision	Access	Cost	Customer

Figure 8.5 ATM classes extracted from problem statement nouns

Communications line	Transaction log

Figure 8.6 ATM classes identified from knowledge of problem domain

| Software | Banking network | Cashier | ATM | Consortium | Bank |

| Bank computer | Account | Transaction | Cashier station | Account data | Transaction data |

| Central computer | Cash Card | User | Cash | Receipt | System |

| Recordkeeping provision | Security provision | Access | Cost | Customer |

Figure 8.5 ATM classes extracted from problem statement nouns

| Communications line | Transaction log |

Figure 8.6 ATM classes identified from knowledge of problem domain

Bad Classes

vague
- System
- Security Provision
- Recordkeeping Provision
- Banking Network

attribute
- Account Data
- Receipt
- Cash
- Transaction Data

redundant
- User

irrelevant
- Cost

implementation
- Transaction Log
- Access
- Software
- Communications Line

Good Classes

| Account | ATM | Bank | Bank Computer | Cash Card | Cashier |

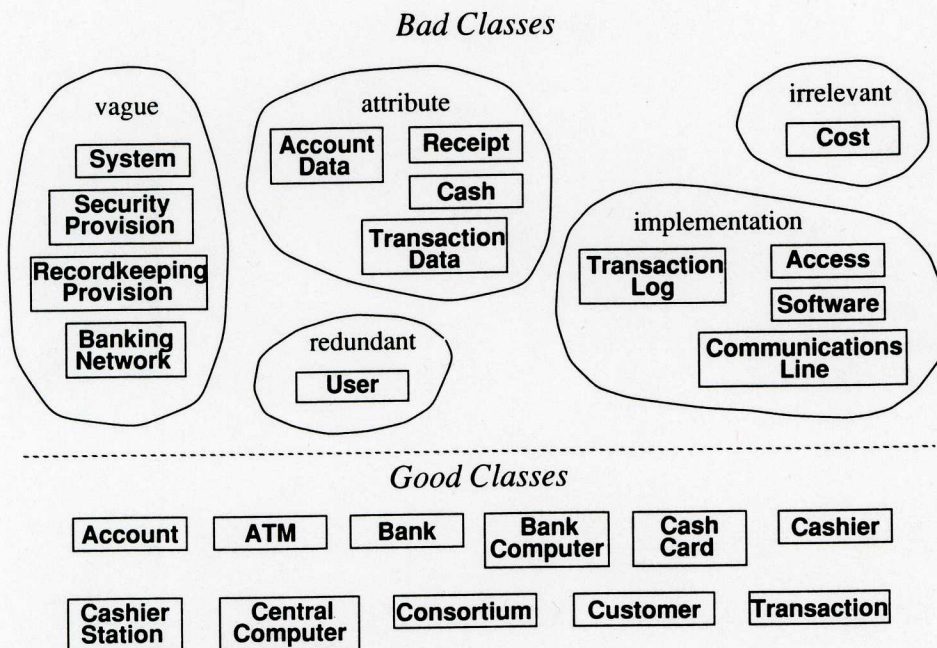| Cashier Station | Central Computer | Consortium | Customer | Transaction |

Figure 8.7 Eliminating unnecessary classes from ATM problem

Account—a single account in a bank against which transactions can be applied. Accounts may be of various types, at least checking or savings. A customer can hold more than one account.

ATM—a station that allows customers to enter their own transactions using cash cards as identification. The ATM interacts with the customer to gather transaction information, sends the transaction information to the central computer for validation and processing, and dispenses cash to the user. We assume that an ATM need not operate independently of the network.

Bank—a financial institution that holds accounts for customers and that issues cash cards authorizing access to accounts over the ATM network.

Bank computer—the computer owned by a bank that interfaces with the ATM network and the bank's own cashier stations. A bank may actually have its own internal network of computers to process accounts, but we are only concerned with the one that talks to the network.

Cash card—a card assigned to a bank customer that authorizes access of accounts using an ATM machine. Each card contains a bank code and a card number, most likely coded in accordance with national standards on credit cards and cash cards. The bank code uniquely identifies the bank within the consortium. The card number determines the accounts that the card can access. A card does not necessarily access all of a customer's accounts. Each cash card is owned by a single customer, but multiple copies of it may exist, so the possibility of simultaneous use of the same card from different machines must be considered.

Cashier—an employee of a bank who is authorized to enter transactions into cashier stations and accept and dispense cash and checks to customers. Transactions, cash, and checks handled by each cashier must be logged and properly accounted for.

Cashier station—a station on which cashiers enter transactions for customers. Cashiers dispense and accept cash and checks; the station prints receipts. The cashier station communicates with the bank computer to validate and process the transactions.

Central computer—a computer operated by the consortium which dispatches transactions between the ATMs and the bank computers. The central computer validates bank codes but does not process transactions directly.

Consortium—an organization of banks that commissions and operates the ATM network. The network only handles transactions for banks in the consortium.

Customer—the holder of one or more accounts in a bank. A customer can consist of one or more persons or corporations; the correspondence is not relevant to this problem. The same person holding an account at a different bank is considered a different customer.

Transaction—a single integral request for operations on the accounts of a single customer. We only specified that ATMs must dispense cash, but we should not preclude the possibility of printing checks or accepting cash or checks. We may also want to provide the flexibility to operate on accounts of different customers, although it is not required yet. The different operations must balance properly.

Figure 8.8 Data dictionary for ATM classes

Verb phrases:

Banking network includes cashiers and ATMs
Consortium shares ATMs
Bank provides bank computer
Bank computer maintains accounts
Bank computer processes transaction against account
Bank owns cashier station
Cashier station communicates with bank computer
Cashier enters transaction for account
ATMs communicate with central computer about transaction
Central computer clears transaction with bank
ATM accepts cash card
ATM interacts with user
ATM dispenses cash
ATM prints receipts
System handles concurrent access
Banks provide software
Cost apportioned to banks

Implicit verb phrases:

Consortium consists of banks
Bank holds account
Consortium owns central computer
System provides recordkeeping
System provides security
Customers have cash cards

Knowledge of problem domain:

Cash card accesses accounts
Bank employs cashiers

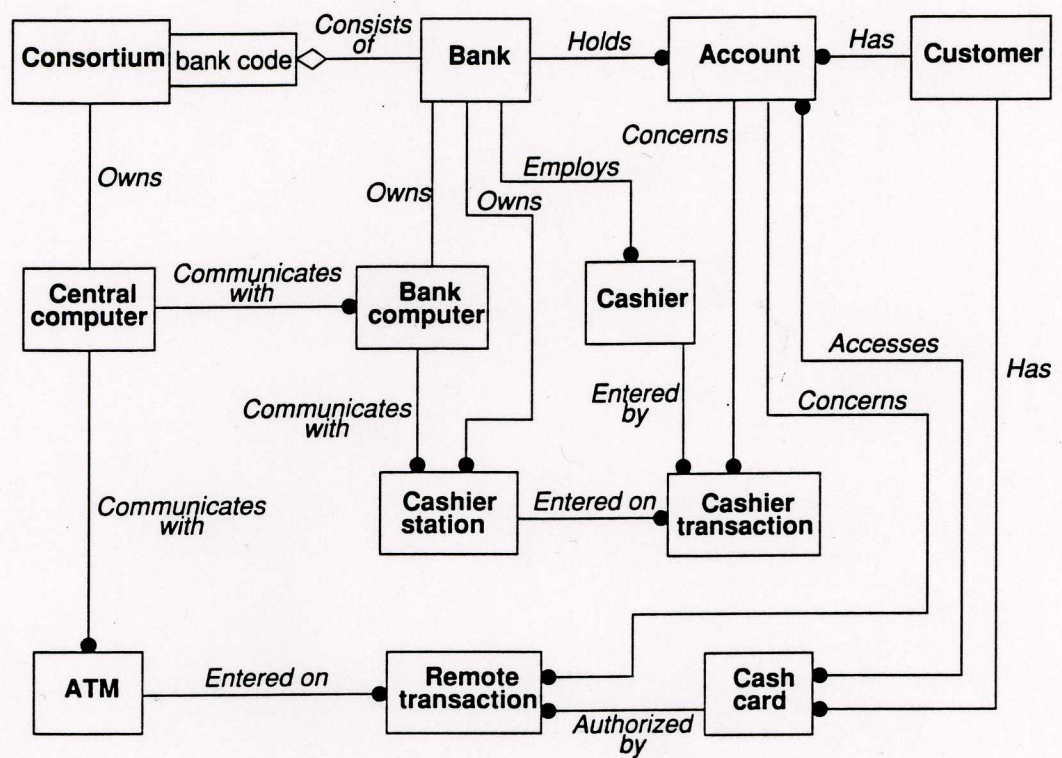**Figure 8.9** Associations from ATM problem statement

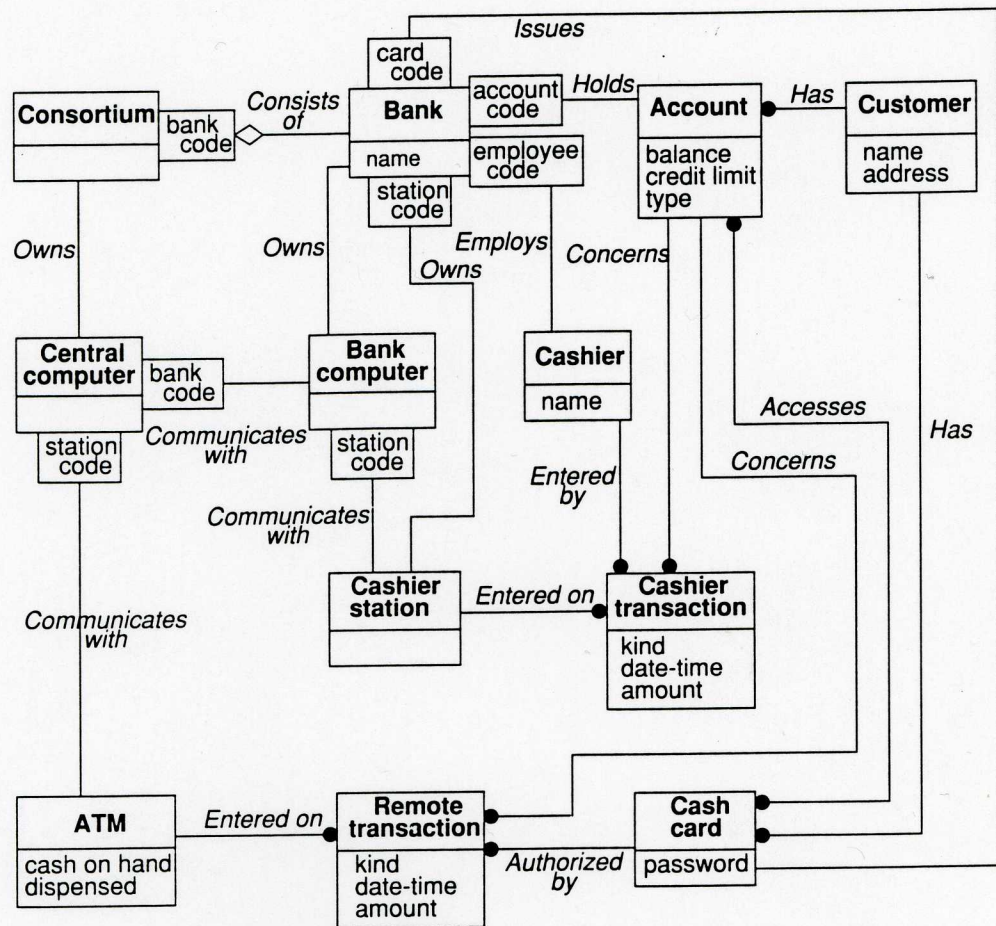**Figure 8.11** Initial object diagram for ATM system
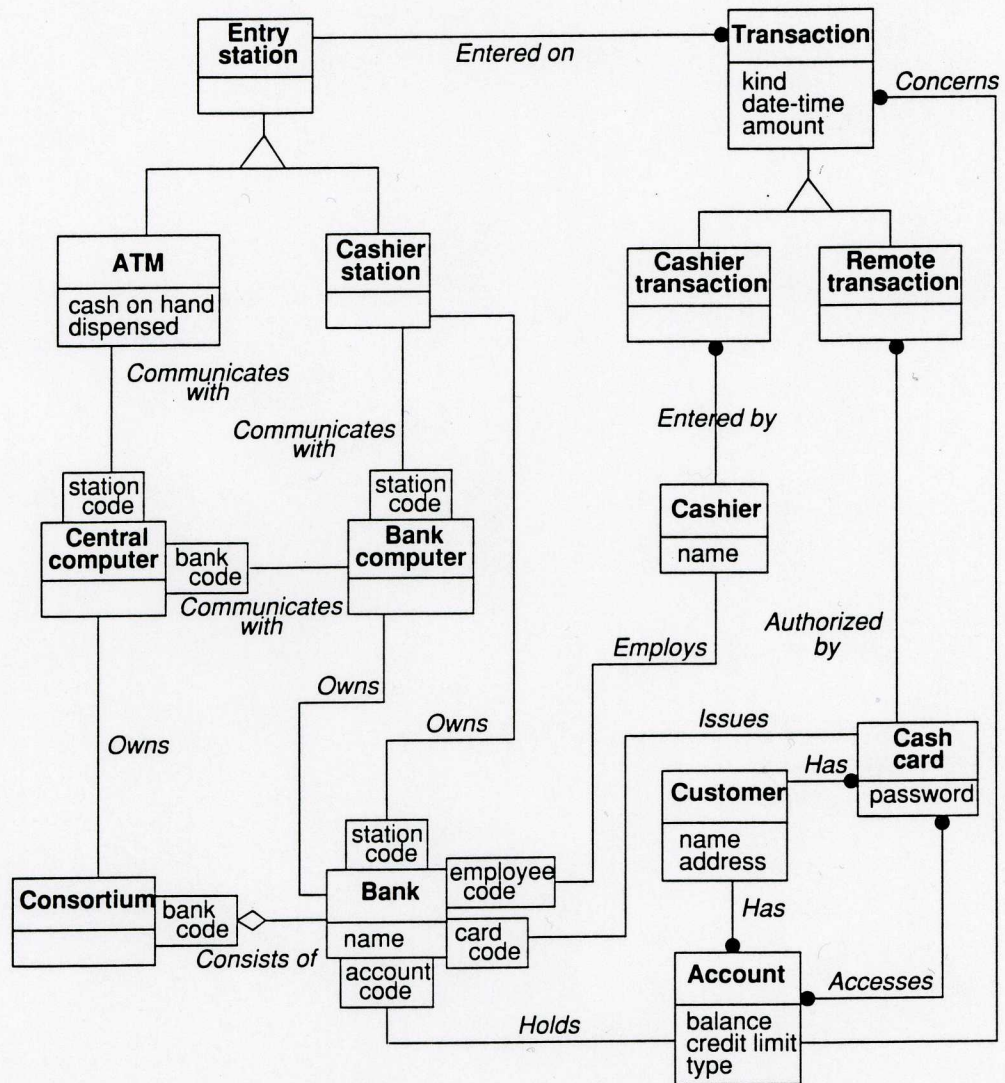
Figure 8.12 ATM object model with attributes

Figure 8.13 ATM object model with attributes and inheritance