2.2 Análisis de Objetos

En esta sección se describirá el proceso de análisis para el modelo de objetos. El primer paso es generar la *descripción del problema.* Como se ha mencionado anteriormente, en la mayoría de los problemas el modelo de objetos es el más importante.

Pasos a seguir para el desarrollo del modelo de objetos:
1) Descripción del problema
2) Elaboración inicial del modelo
3) Verificación del modelo
4) Iteración del modelo
5) Diccionario de Datos
6) Agrupación en Módulos

Los pasos a seguir en el análisis sobre el modelo de objetos no son necesariamente secuenciales, pudiendo ocurrir de forma concurrente.

2.2.1 Descripción del Problema

La descripción del problema es una descripción inicial, la cual sirve como punto de comienzo para comprender los requisitos del sistema.

- La descripción del problema debe ser una descripción de necesidades y no una propuesta para una solución. La descripción del problema es una especificación de requisitos.

- La descripción inicial puede ser incompleta e informal, la cual se hace más precisa durante el análisis, exponiendo sus ambigüedades e inconsistencias.

- El propósito del análisis es comprender completamente el problema y sus implicaciones. No hay razón para esperar que la descripción inicial del problema, preparado sin un análisis completo, sea correcto.

- El análisis no es un proceso mecánico, sino que el analista debe comunicarse con el cliente de forma interactiva para completar la información faltante, y para clarificar ambigüedades e inconsistencias.

- El analista debe separar entre los requisitos verdaderos, y las decisiones relacionadas al diseño e implementación. Se debe indicar cuales aspectos son mandatorios y cuales aspectos son opcionales, para evitar restringir la flexibilidad de la implementación. Los diseños subsecuentes se refieren al modelo de análisis y no a la descripción del problema.

- Durante el diseño se debe incluir especificaciones de rendimiento y protocolos de interacción con sistemas externos, al igual que provisiones sobre modularidad y futuras extensiones.

- En ciertas ocasiones se puede incluir en la especificación del sistema aspectos de diseño, como el uso de lenguajes de programación particulares.

- Como regla general, si se hace exactamente lo que el cliente desea pero el resultado no es satisfactorio, el diseñador probablemente sea culpado.

Ejemplo: Un diagrama describiendo un sistema de *Cajeros* (*cajeros humanos* y *máquina de cajero automático* (ATM: Automatic Teller Machine)) se muestra en la Figura 2.139. (Este diagrama no parte de la descripción del problema.)

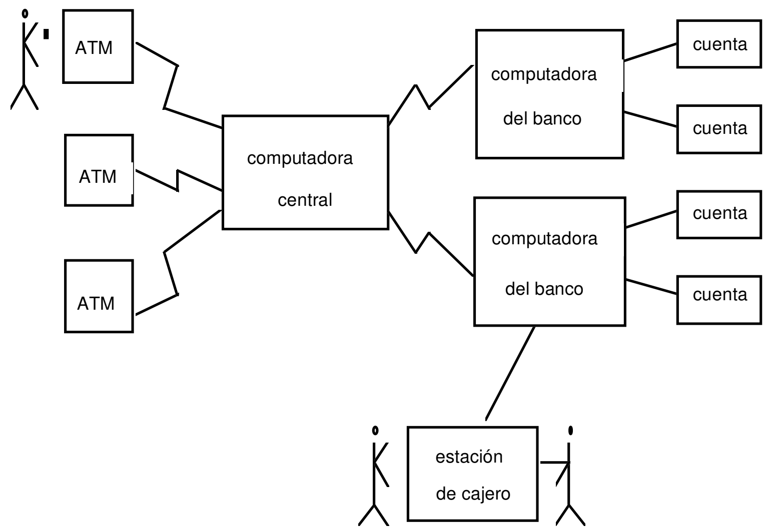

Figura 2.139. Diagrama general del sistema para una red de cajeros, incluyendo cajeros automáticos (ATM) y estaciones de cajeros humanos.

Ejemplo: El texto para la *Descripción del Problema* de un sistema de *Cajeros*, suministrado por el cliente, se muestra a continuación:

- Diseñar el software para una red computarizada del banco que incluye estaciones de cajeros humanos y máquinas de cajeros automáticos (ATM) para ser compartidos por un consorcio de bancos.
- Cada banco posee su propia computadora para mantener sus propias cuentas y procesos de transacción.
- Los bancos poseen las estaciones de cajero que se comunican directamente con las computadoras del banco.
- Los cajeros humanos entran cuentas y datos de transacción.
- Los ATM se comunican con la computadora central que verifica las transacciones con el banco apropiado.
- Los ATM aceptan tarjetas de cajero automático, interaccionan con el usuario, se comunican con el sistema central para llevar a cabo las transacciones, retiran efectivo, depositan, e imprimen recibos.
- El sistema requiere registros apropiados y provisiones de seguridad.
- El sistema debe manejar correctamente accesos concurrentes a la misma cuenta.

- El banco debe proveer su propio software para sus computadoras.
- El costo del sistema compartido se divide entre los bancos según el número de clientes con tarjetas de cajero automático.

2.2.2 Elaboración Inicial del Modelo

La elaboración inicial del modelo consiste en analizar los aspectos básicos para la construcción del diagrama de clases. Esto incluirá:
1) Clases
2) Asociaciones y Agregaciones
3) Atributos
Para ello se seguirán los siguientes pasos:
a) Identificación
b) Selección
c) Refinamiento
Finalmente se incorporará:
4) Multiplicidad
5) Herencia

2.2.2.1 Clases

Se debe determinar las clases, explícitas o implícitas, en la descripción del problema. El proceso a seguir se muestra en la Figura 2.140.

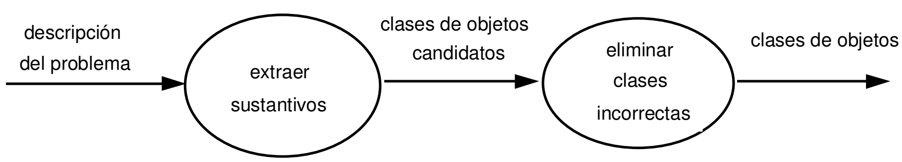

Figura 2.140. Diagrama de identificación de clases de objetos.

2.2.2.1.1 Identificación de Clases

Para la identificación de clases se extraen todos los sustantivos de la descripción del problema.

- Los sustantivos en la descripción del problema son los posibles candidatos a clases de objetos.

Ejemplo: "Un sistema de reservaciones que vende boletos para funciones a varios teatros"
Las clases candidatas son: *Sistema de Reservaciones*, *Boletos*, *Función*, *Teatro*.

- Durante esta etapa, se debe identificar entidades físicas, al igual que entidades conceptuales.

- No se debe tratar de diferenciar entre clases y atributos durante ésta etapa.

- No todas las clases son explícitas en la descripción del problema, algunas son implícitas en la aplicación, siendo necesario añadir clases que pueden ser identificadas por nuestro conocimiento del área.

- Se debe revisar los pronombres en la descripción del problema para asegurar que no se haya perdido ningún sustantivo descrito de forma implícita.

- Para facilitar la identificación de clases, se subrayan todos los sustantivos de la descripción del problema.

Ejemplo Cajeros:

- Diseñar el software para una red computarizada del banco que incluye estaciones de cajeros humanos y máquinas de cajeros automáticos (ATM) para ser compartidos por un consorcio de bancos.
- Cada banco provee su propia computadora para mantener sus propias cuentas y procesos de transacción.
- Las estaciones de cajero pertenecen a los bancos individuales y se comunican directamente con sus propias computadoras del banco.
- Los cajeros humanos entran cuentas y datos de transacción.
- Los ATM se comunican con la computadora central que verifica las transacciones con el banco apropiado.
- Los ATM aceptan tarjetas de cajero automático, interaccionan con el usuario, se comunican con el sistema central para llevar a cabo las transacciones, retiran efectivo, depositan, e imprimen recibos.
- El sistema requiere registros apropiados y provisiones de seguridad.
- El sistema debe manejar correctamente accesos concurrentes a la misma cuenta.
- El banco debe proveer su propio software para sus computadoras.
- El costo del sistema compartido se divide entre los bancos según el número de clientes con tarjetas de cajero automático.

En el sistema de cajeros, los sustantivos encontrados en la descripción del problema son los siguientes (en singular):
1) *software*
2) *red computarizada del banco*
3) *estación de cajero*
4) *ATM*
5) *consorcio de bancos*
6) *banco*
7) *computadora*
8) *cuenta*
9) *proceso de transacción*
10) *computadora del banco*
11) *cajero humano*
12) *dato de transacción*

13) *computadora central*
14) *tarjeta de cajero automático*
15) *usuario*
16) *sistema central*
17) *transacciones*
18) *efectivo*
19) *recibo*
20) *registro*
21) *provisión de seguridad*
22) *acceso concurrentes*
23) *costo del sistema*
24) *cliente*

2.2.2.1.2 Selección de Clases

Partiendo de todas las clases identificadas según la descripción del problema, se debe seleccionar cuales son las clases importantes para el análisis.

- Todas las clases deben tener sentido en el área de la aplicación, la relevancia al problema debe ser el único criterio para la selección.

- Como regla general, se debe escoger los nombres para las clases con cuidado; para que estos no sean ambiguos. Este es uno de los procesos más difíciles. Los nombres deben ser establecidos con un formato consistente (e.g. nombres en plural o singular).

- No hay que preocuparse durante esta etapa sobre asociación, agregación, o herencia. Primero hay que tener las clases específicas correctas para no suprimir detalles en el intento de ajustarse a estructuras preconcebidas.

- Se deben eliminar las clases redundantes, si las dos expresan la misma información. La clase más descriptiva debe ser guardada.

Ejemplo Cajero: *Cliente* y *Usuario* son redundantes. *Usuario* puede ser mas descriptivo para una aplicación de computación. En el caso del ATM, *Cliente* es más descriptivo y se guarda.

- Se deben eliminar las clases irrelevantes, que tienen poco o nada que ver con el problema. Esto requiere juicio porque en un contexto una clase puede ser importante mientras que en otro contexto la clase podría no serlo.

Ejemplo Cajero: *Costo del Sistema* está fuera del alcance de las transacciones del sistema, aunque sea importante para dividir las cuentas del consorcio.

- Se deben eliminar las clases que debieran ser atributos más que clases, cuando los nombres corresponden a propiedades más que entidades independientes.

Ejemplo Cajero: *dato de cuenta* podría definirse como un atributo de la clase *cuenta*. Un ATM da *efectivo* y *recibos* que no afectan de otra manera al problema, y deberían ser incluidos como atributos. También *dato de transacción* debería ser incluido como atributo de *transacción*.

- Se deben eliminar las clases que debieran ser operaciones más que clases, si las entidades representan operaciones que son aplicadas a los objetos y no entidades manipuladas por si mismas.

Ejemplo: *Proceso de Transacción* es una operación que se aplica a la *Transacción*, y no existe como una entidad independiente con sus propios atributos. En este caso los atributos pertenecerían a *Transacción*.

- Se deben eliminar las clases que debieran ser roles, si el nombre describe el rol que la clase juega en una asociación y no su naturaleza.

Ejemplo: *Dueño* sería un mal nombre para una clase en una base de datos de un fabricante de coches; ya que si se añade personas que arriendan (*lease*) un coche, la relación sería incorrecta. El nombre correcto sería *Persona* (o cliente), que asume varios roles diferentes: *dueño* o *arrendador*.

- Se deben eliminar las clases que corresponden a construcciones de implementación si están alejadas del mundo real, por lo cual deben ser eliminadas del análisis. Pueden ser necesarias luego durante el diseño.

Ejemplo: *subrutinas*, *listas, y arreglos*, son clases típicas de implementación.

Ejemplo Cajero: La representación de *Registros* (de *Transacción*) es un aspecto del diseño. *Línea de Comunicación* es una implementación física de una asociación de comunicación entre diferentes objetos de computación. *Acceso Concurrente* y *Software* también son entidades de implementación.

- Se deben clarificar las clases imprecisas, si describen de forma no específica su función. Algunas clases pueden tener bordes mal definidos o demasiado generales.

Ejemplo Cajero: *Sistema Central*, *Provisión de Seguridad*, *Computadora*, y *Red Computarizada del Banco* son clases imprecisas. *Estaciones de Cajero* es imprecisa ya que hay cajeros humanos y automáticos. *Transacción* es una clase imprecisa ya que debería repartirse en *Transacciones de Cajero Humano* y *Transacciones de Cajero Automático*: Durante la introducción de herencia puede que sea necesario una clase para compartir aspectos comunes a ambas clases.

- Ante la duda, se deben conservar las clases, ya que posteriormente siempre habrá oportunidad para eliminarlas.

La siguiente tabla muestra las razones para la eliminación de sustantivos de la descripción del problema:

SUSTANTIVOS	ELIMINACION
software	construcción de implementación
red computadora del banco	construcción de implementación
estación de cajero (humano)	
máquina de cajero automático	
consorcio de bancos	
banco	
computadora	impreciso
cuenta	
proceso de transacción	operación
computadora de banco	
cajero humano	
datos de transacción	atributo
computadora central	
tarjeta de cajero automático	
usuario	redundante (= cliente)
sistema central	impreciso
transacción	
efectivo	atributo
recibo	atributo
registro	construcción de implementación
provisión de seguridad	impreciso
acceso concurrente	construcción de implementación
costo del sistema	irrelevante
cliente	

Las siguientes son las clases que quedarían para el sistema ATM:
1) *estación de cajero humanos*
2) *máquina de cajero automático*
3) *consorcio de bancos*
4) *banco*
5) *cuenta*
6) *computadora del banco*
7) *cajero humano*
8) *computadora central*
9) *tarjeta de cajero automático*
10) *transacción*
11) *cliente*

Se incluye *consorcio de bancos* aunque sea único, ya que es una parte fundamental del problema, y puede ser útil para futuras expansiones donde existan múltiples consorcios.

2.2.2.1.3 Diagrama de Clases

Después de haber identificado las clases durante el análisis, se debe construir la versión inicial del diagrama de clases.

Ejemplo Cajero: El diagrama de clases se muestra en la Figura 2.141 (el orden interno dentro del diagrama es ubicar clases posiblemente relacionadas cerca una de la otra).

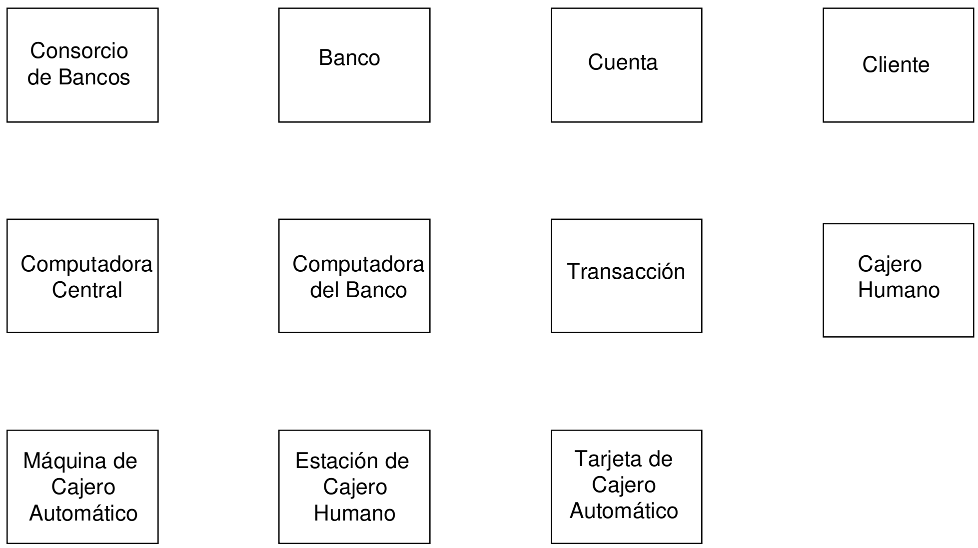

Figura 2.141. Diagrama de clases para el sistema de cajeros.

- El diagrama inicial de clases puede ayudar a identificar clases que estén ausentes, y junto con el diccionario de datos, puede ayudar a generalizar las clases similares.

Ejemplo: Ya podemos ir viendo que los dos tipos de transacciones y los dos tipos de estaciones tienen características comunes.

2.2.2.1.4 Refinamiento de Clases

A continuación se puede hacer un refinamiento sobre el diagrama de clases. Opcionalmente se puede continuar con el análisis del modelo de objetos para posteriormente refinar las clases, como será hecho con el ejemplo actual del *Sistema de Cajeros*.

2.2.2.2 Asociaciones y Agregaciones

Se debe determinar las asociaciones y agregaciones, explícitas o implícitas, en la descripción del problema.

2.2.2.2.1 Identificación de Asociaciones y Agregaciones

Las asociaciones y agregaciones corresponden a referencias de una clase a otra (u otras).

- Las asociaciones y agregaciones son expresadas como verbos y frases verbales en la descripción del problema (descripción inicial). No se debe limitar a las clases ya seleccionadas anteriormente, sino a todos los sustantivos (y pronombres) encontrados originalmente en la descripción del problema. Se hace esto para evitar omitir una asociación importante sobre una clase que pudo haber sido eliminada por error.

- Los verbos o frases verbales pueden incluir:
  - lugar físico (e.g. *al-lado-de*, *parte-de*, *contenido-en*),
  - acciones dirigidas (e.g. *maneja*),
  - comunicación (e.g. *habla-con*),
  - propiedad (e.g. *tiene*, *parte-de*),
  - puede satisfacer alguna condición (e.g. *trabaja-para*, *casado-con*).

- Para algunas asociaciones la frase verbal es implícita en la oración, mientras que otras dependen del conocimiento del mundo real o de suposiciones.

- La agregación es una forma especial de asociación relacionando dos o más clases. Inicialmente no es necesario distinguir entre asociaciones y agregaciones.

Ejemplo Cajero: Algunas de las siguientes asociaciones son ternarias aunque pudieran haberse definido como varias binarias también. Así mismo, podríamos haber encontrado un número mayor de asociaciones ternarias.

<u>Asociaciones explícitas:</u>

1) *El software se diseña*
2) *La red computarizada del banco incluye estaciones de cajeros humanos y ATM*
3) *El consorcio comparte los ATM*
4) *Cada banco posee su propia computadora*
5) *La computadora del banco mantiene sus propias cuentas*
6) *La computadora del banco mantiene sus procesos de transacción*
7) *Los bancos poseen las estaciones de cajero*
8) *Las estaciones de cajero humano se comunican con sus propias computadoras del banco*
9) *Los cajeros humanos entran las cuentas*
10) *Los cajeros humanos entran los datos de transacción*
11) *Los ATM se comunican con la computadora central*
12) *La computadora central verifica las transacciones con el banco*
13) *Los ATM aceptan tarjetas de cajero automático*
14) *Los ATM interaccionan con el usuario*
15) *Los ATM se comunican con el sistema central*
16) *El sistema central lleva a cabo las transacciones*
17) *Los ATM retiran efectivo*
18) *Los ATM depositan*
19) *Los ATM imprimen recibos*
20) *El sistema requiere registros apropiados*

21) *El sistema requiere provisiones de seguridad*
22) *El sistema maneja acceso concurrentes a una cuenta*
23) *El banco provee el software para sus computadoras*
24) *El costo del sistema es proporcional a los bancos*

Asociaciones Implícitas:

25) *El consorcio consiste de bancos*
26) *El banco posee las cuentas*
27) *El consorcio posee una computadora central*
28) *El sistema provee los registros*
29) *El sistema requiere seguridad*
30) *Los clientes tienen las tarjetas de cajero automático*
31) *Las tarjetas de cajero automático accesan las cuentas de los clientes*
32) *El banco emplea a los cajeros*

2.2.2.2.2 Selección de Asociaciones y Agregaciones

Se deben eliminar las asociaciones innecesarias e incorrectas según los siguiente criterios (algunas frases pueden ser eliminadas por múltiples razones):

- Se deben eliminar las asociaciones entre clases eliminadas. Si una de las clases en la asociación fue eliminada, la asociación debe ser eliminada o expresada en término de otras clases.

Ejemplo Cajero: las asociaciones eliminadas son:
el software se diseña,
la red computarizada del banco incluye estaciones de cajeros humanos y ATM,
los cajeros humanos entran los datos de transacción,
los ATM se comunican con el sistema central,
el sistema central lleva a cabo las transacciones,
los ATM retiran efectivo, los ATM depositan,
los ATM imprimen recibos,
el sistema requiere registros apropiados,
el sistema requiere provisiones de seguridad,
el sistema maneja acceso concurrente a una cuenta,
el banco provee el software para sus computadoras,
el costo del sistema es proporcional a los bancos.

- Se deben eliminar las asociaciones de implementación, las asociaciones que especifiquen construcciones de implementación.

Ejemplo Cajero: las asociaciones eliminadas por conceptos de implementación son:
el sistema maneja el acceso concurrente,
el sistema provee los registros.

- Se deben eliminar las asociaciones irrelevantes que no sean importantes para la solución del problema. Son importantes las asociaciones que afectan entradas o salidas, u objetos dentro del sistema.

Ejemplo Cajero: Se puede eliminar:
el costo del sistema es proporcional a los bancos,
el sistema provee la seguridad.
(El costo y la seguridad son irrelevantes al problema.)

- Se deben eliminar las asociaciones derivadas o redundantes que están definidas en término de otras asociaciones. Se deben guardar las asociaciones más generales.

Las razones para la eliminación de las frases explícitas se muestra en la siguiente tabla:

VERBOS	ELIMINACION
El software se diseña	clases eliminadas
La red computarizada del banco incluye estaciones de cajeros y ATM	clases eliminadas
El consorcio comparte los ATM	
Cada banco posee su propia computadora	
La computadora del banco mantiene sus propias cuentas	
La computadora del banco mantiene sus procesos de transacción	clases eliminadas
Los bancos poseen las estaciones de cajero	
Las estaciones de cajeros se comunican con las computadoras del banco	
Los cajeros humanos entran las cuentas	
Los cajeros humanos entran los datos de transacción	clases eliminadas
Los ATM se comunican con la computadora central	
La computadora central verifica las transacciones con el banco	
Los ATM aceptan tarjetas de cajero automático	
Los ATM interaccionan con el usuario	
Los ATM se comunican con el sistema central	redundante
El sistema central lleva a cabo las transacciones	clases eliminadas
Los ATM retiran efectivo	clases eliminadas
Los ATM depositan	clases eliminadas
Los ATM imprimen recibos	clases eliminadas
El sistema requiere registros apropiados	implementación
El sistema requiere provisiones de seguridad	irrelevante
El sistema maneja acceso concurrentes a una cuenta	implementación
El banco provee el software para sus computadoras	clases eliminadas
El costo del sistema es proporcional a los bancos	irrelevante

Las razones para la eliminación de las frases implícitas se muestra en la siguiente tabla:

FRASES	ELIMINACION
El consorcio consiste de bancos	
El banco posee las cuentas	

El consorcio posee una computadora central El sistema provee los registros El sistema provee la seguridad Los clientes tienen las tarjetas de cajero automático Las tarjetas de cajero automático accesan las cuentas de los clientes El banco emplea a los cajeros	implementación irrelevante

2.2.2.2.3 Diagrama de Clases con Asociaciones

Después de haber identificado las asociaciones iniciales durante el análisis, se debe construir la versión inicial del diagrama de clases incluyendo asociaciones.

Ejemplo Cajero: El diagrama de clases con asociaciones se muestra en la Figura 2.142.

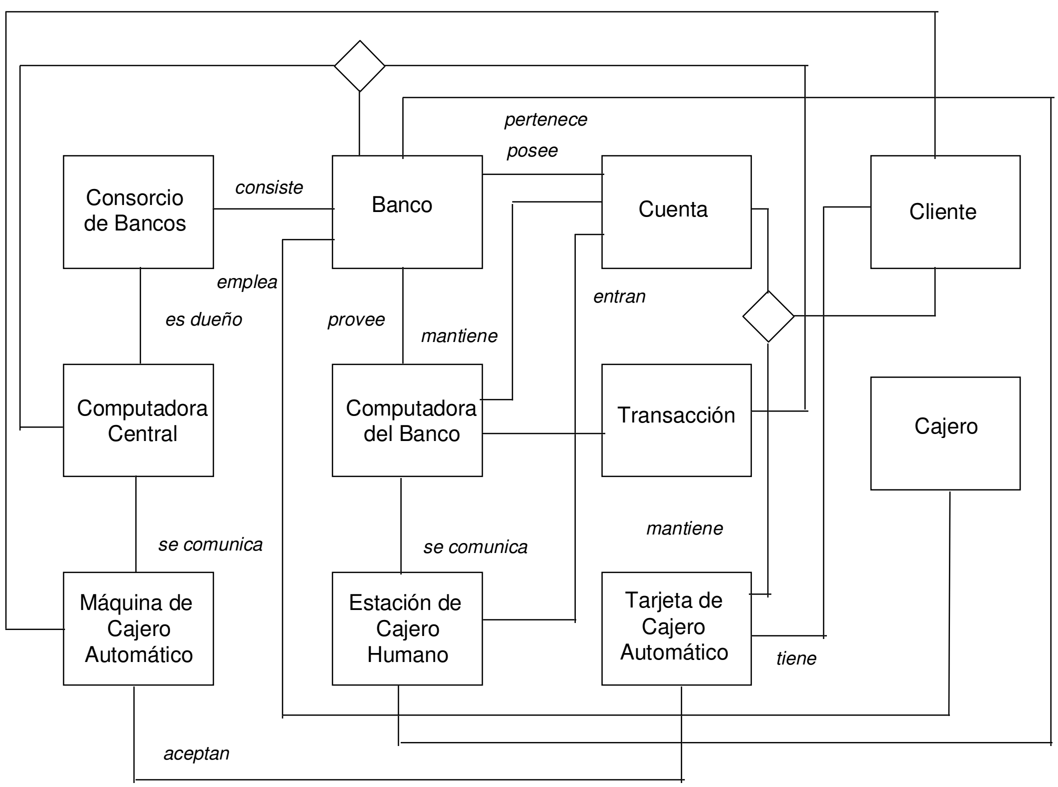

Figura 2.142. Diagrama de clases con asociaciones para el sistema de cajeros.

2.2.2.2.4 Refinamiento de Asociaciones

Luego de completado el primer diagrama de clases con asociaciones, se debe revisar las asociaciones encontradas para ver si estas son realmente necesarias, en particular ver si los nombres están bien escogidos.

- Se debe describir la asociación y no su razón de ser.

- Las operaciones no son asociaciones, sólo indicadores de que una asociación debe existir.

- Operaciones múltiples entre dos clases no requieren múltiples asociaciones, con una sola puede que sea suficiente.

Ejemplo Cajero: las asociaciones modificadas son:
la computadora del banco mantiene las cuentas,
es una expresión de acción. Se debería llamar:
el banco posee cuentas.

- Una asociación debe describir una propiedad estructural del dominio de la aplicación, y no un evento transitorio.

Ejemplo Cajero:
la máquina de cajero automático acepta las tarjetas de cajero automático
describe parte de la interacción entre
máquina de cajero automático y *cliente*.
No es una relación permanente entre
máquina de cajero automático y *tarjetas de cajero automático*.
También se puede eliminar:
máquina de cajero automático interacciona con el cliente.

- A veces un requisito expresado como una acción implica una relación estructural y debería ser re-expresada de forma correspondiente.

Ejemplo Cajero:
La computadora central verifica las transacciones con el banco
describe una acción que implica una relación estructural
la computadora central se comunica con el banco para las transacciones.

El diagrama del sistema de cajeros, según las modificaciones anteriores se muestra en la Figura 2.143.

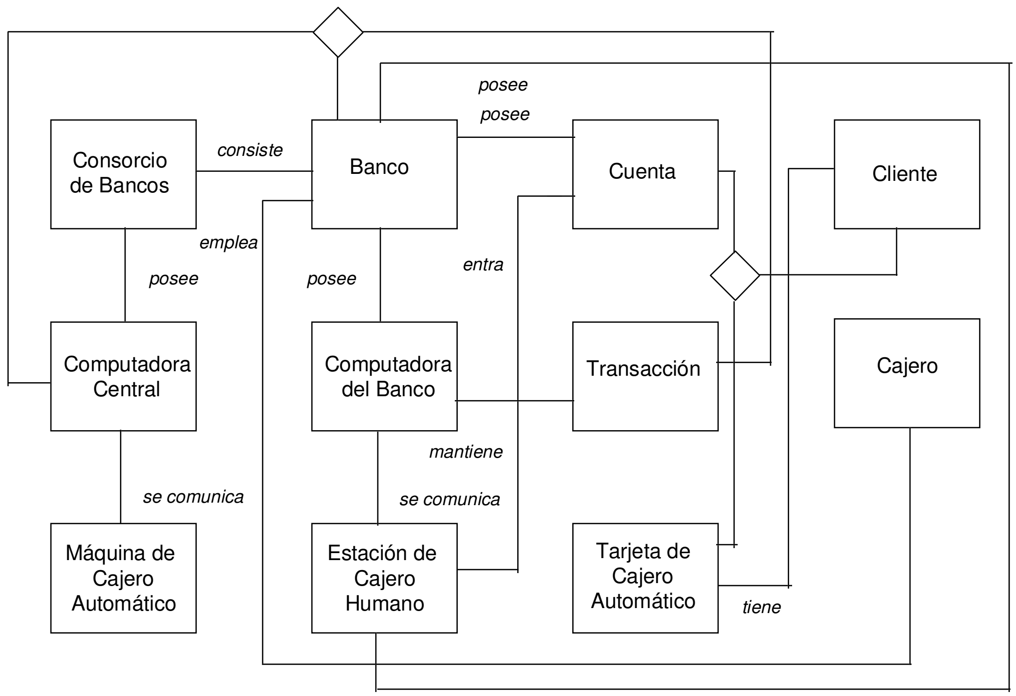

Figura 2.143. Diagrama de clases para el sistema de cajeros con nuevos nombres de asociación.

2.2.2.2.5 Descomponer Asociaciones de Alto Grado

La mayoría de las relaciones entre tres o más clases se pueden descomponer en asociaciones binarias, aunque no se debe necesariamente descomponer todas las existentes, sólo si tiene sentido la descomposición.

Ejemplo Cajero: *Las tarjetas de cajero automático accesan las cuentas de los clientes* puede separarse en tres asociaciones binarias, *el cliente tiene cuentas, el cliente tiene tarjetas de cajero,* y *las tarjetas de cajero accesan cuentas*.

Ejemplo Cajero: *La computadora central verifica las transacciones con el banco* se pueden separar en un par de asociaciones binarias: *La computadora central se comunica con el banco* y *la transacción es entrada en el ATM*.

El diagrama para el sistema de cajeros con la eliminación de asociaciones ternarias se muestra en la Figura 2.144.

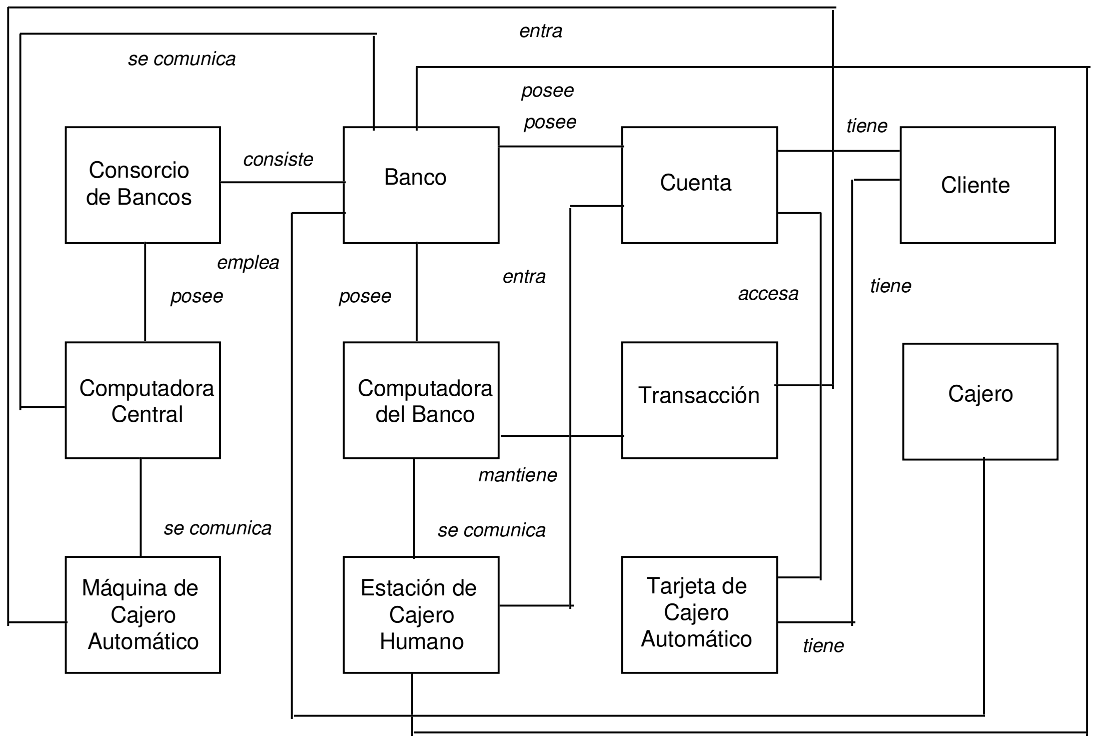

Figura 2.144. Diagrama de clases para el sistema de cajeros con la descomposición de las asociaciones de alto grado.

2.2.2.2.6 Agregar Asociaciones Ausentes

Se deben identificar las asociaciones que por algún motivo hayan sido omitidas, según la descripción inicial del problema y/o por conocimiento general del área.

Ejemplo Cajero: Faltan:
la transacción es entrada en la estación de cajero,
el cajero entra las transacciones,
el banco emplea cajeros,
la transacción concierne cuentas,
la transacción autorizada por la tarjeta de dinero.
(Si los cajeros están restringidos a una sola estación entonces la asociación:
el cajero autorizado en la estación de cajero
sería necesario.)

El nuevo diagrama para el sistema de cajeros se muestra en la Figura 2.145.

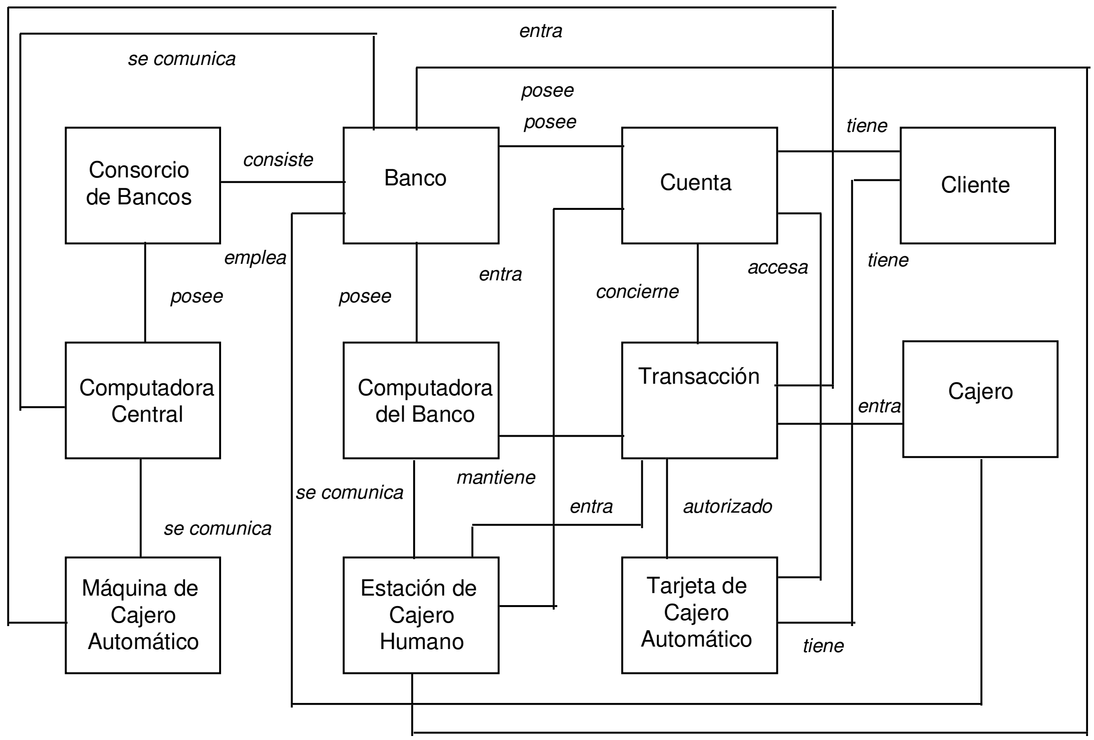

Figura 2.145. Diagrama de clases para el sistema de cajeros añadiendo las asociaciones ausentes.

2.2.2.2.7 Determinar la Multiplicidad

Se debe determinar la multiplicidad para cada asociación. No es necesario dedicar demasiado tiempo a ello, ya que la multiplicidad de la relación puede variar durante el análisis.

El diagrama 2.146 muestra el sistema de cajeros con la incorporación de multiplicidades.

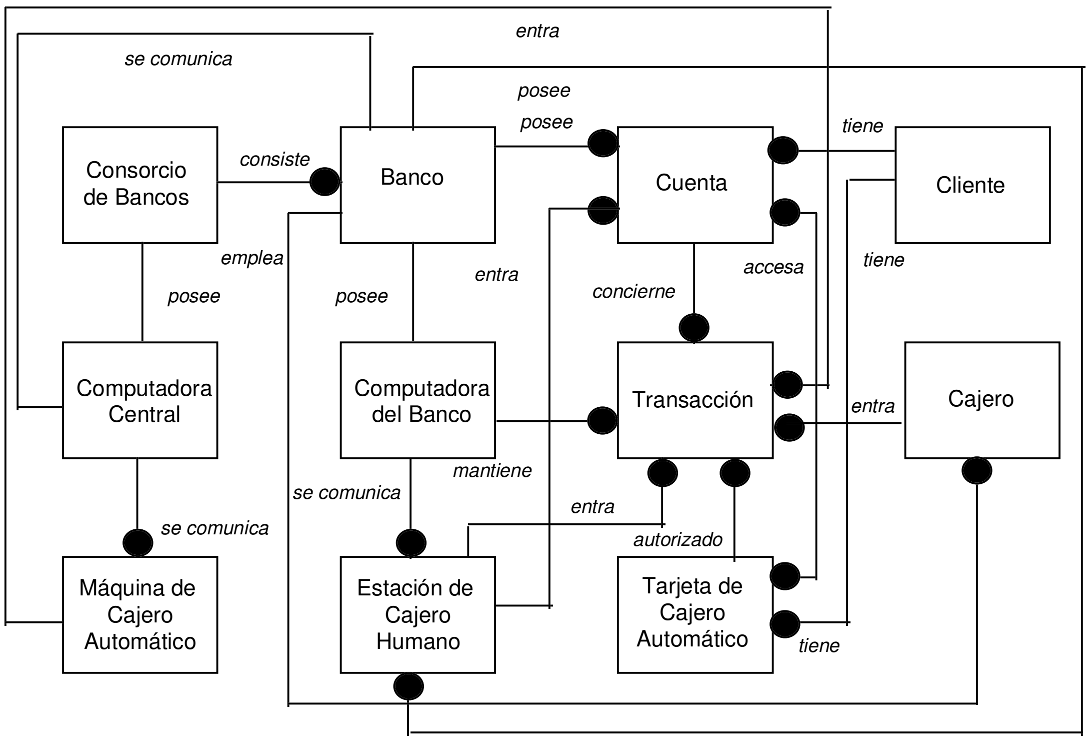

Figura 2.146. Diagrama de clases para el sistema de cajeros con asociaciones y multiplicidad.

2.2.2.2.8 Agregar Nombres de Rol

El nombre del rol describe el papel que una clase juega en una asociación desde el punto de vista de la otra clase.

- Si hay una sola asociación entre un par de clases, y el nombre de la clase describe adecuadamente su rol, se puede omitir los nombres de rol.

Ejemplo Cajero: El rol en la asociación:
computadora central se comunica con las máquinas de cajero automático
está claro en los nombres de la clase.

2.2.2.2.9 Nuevo Refinamiento de Asociaciones

Es necesario seguir iterando este análisis para lograr llegar a un diagrama que satisfaga la descripción del problema.

Ejemplo Cajero: La asociación:
la computadora central se comunica con el banco
podría ser modificada a:
la computadora central se comunica con la computadora del banco.

- Como parte del proceso iterativo se vuelve a eliminar las asociaciones redundantes.

Ejemplo Cajero: La relación *la computadora del banco mantiene las transacciones de cajero* es una asociación redundante, ya que existen relaciones entre *computadora del banco* y *estación de cajero*, y entre *estación de cajero* y *transacción de cajero*. De forma similar, la relación los cajeros humanos entran las cuentas no es necesaria ya que el cajero humano entra transacción, y la transacción concierne a la cuenta. También se puede añadir *El banco otorga tarjetas de cajero automático al cliente*.

El diagrama de clases para el sistema de cajeros se muestra en la Figura 2.147 de acuerdo a los últimos cambios, con una reorganización esquemática del diagrama para presentar las asociaciones de forma más clara.

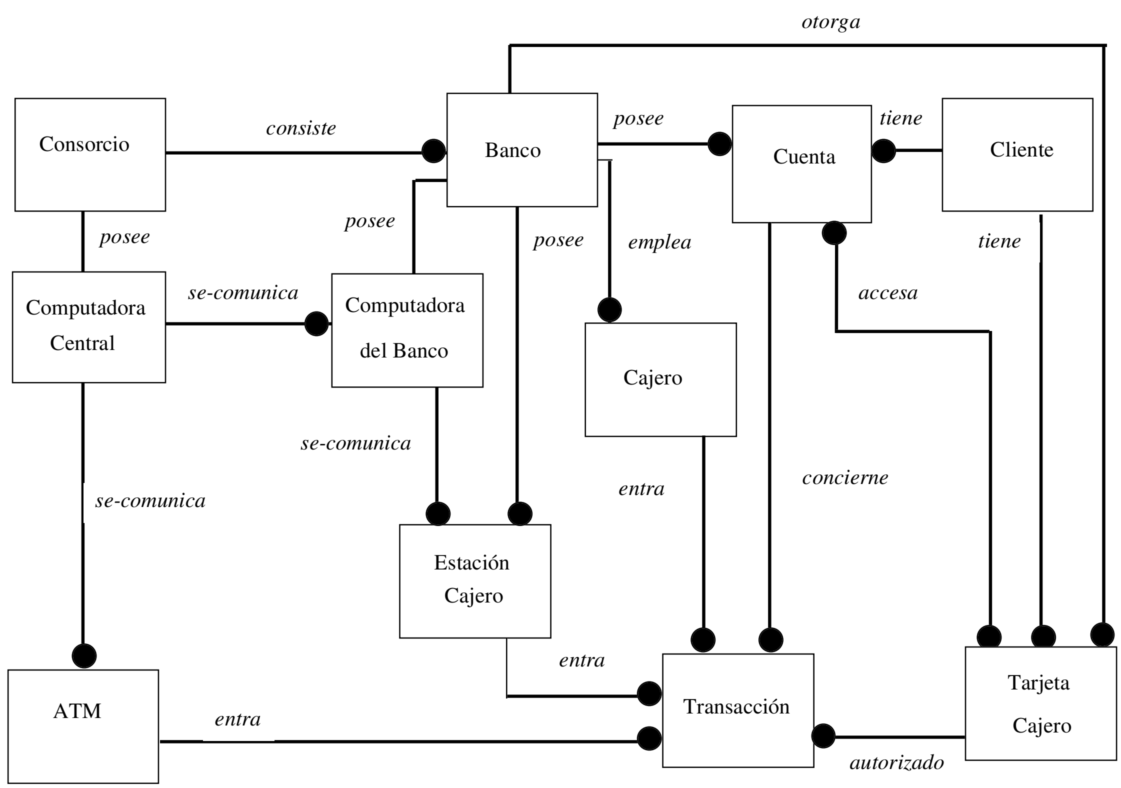

Figura 2.147. Nuevo diagrama de clases para el sistema de cajeros incluyendo asociaciones y multiplicidad.

2.2.2.3 Atributos

Se deben determinar los atributos según la descripción del problema.

2.2.2.3.1 Identificación de Atributos

Ya completado el diagrama de clases para las clases y asociaciones, se debe proseguir con la identificación de los atributos según la descripción del problema.

- A diferencia de las clases y las asociaciones, los atributos tienen menos probabilidad de ser descritos en el problema. La mayoría de los atributos se descubren del conocimiento de la aplicación.

- Los atributos son generalmente nombres seguidos de frases posesivas. Los adjetivos en las frases representan por lo general los valores de los atributos.

Ejemplo Cajero: *Retirar dinero*, *depositar* e *imprimir recibos* son valores enumerados para *tipo de transacción*.

- Algunos atributos se encuentran al eliminar las clases en la lista inicial de sustantivos.

Ejemplo Cajero: Los siguientes son los atributos iniciales encontrados de forma explícita dentro de la descripción del problema, y los implícitos encontrados por conocimiento del problema. (Nótese que muchos de los atributos implícitos podrán ser eliminados al analizar mejor el problema.)

Atributos explícitos
1) *datos de transacción* (*tipo: retiro, depósito, Fecha-Hora, Cantidad*)
2) *tipo de transacción* (*retiro, depósito, imprimir recibo*)

Atributos implícitos
3) *datos de cuenta* (*balance, crédito límite, tipo: cheque, ahorro*)
4) *código de cuenta*
5) *identificador de transacción*
6) *nombre de cliente*
7) *dirección de cliente*
8) *código de cajero*
9) *nombre de cajero*
10) *código de banco*
11) *nombre de banco*
12) *código de estación de cajero*
13) *código de máquina de cajero automático*
14) *datos de máquina de cajero automático* (*hay dinero, dinero dado*)
15) *código de tarjeta de cajero*
16) *contraseña de tarjeta de cajero*

2.2.2.3.2 Selección de Atributos

Se deben eliminar los atributos incorrectos o innecesarios.

- Si la existencia independiente de una entidad es importante, más que un simple valor, entonces la entidad debe definirse como una clase.

- Se deben evitar los atributos de implementación. No se debe listar identificadores en el modelo de objeto, ya que estos son implícitos en el sistema. Solo se deben listar atributos que existen en la aplicación.

Ejemplo Cajero: *código de cuenta* es un buen atributo, ya que el *banco* asigna *códigos de cuenta*. En contraste, identificadores de *transacción* internos del sistema no deberían ser listados como atributo, aunque sea conveniente generar uno durante la implementación.

- Si un atributo describe el estado interno de un objeto que es invisible fuera del objeto, entonces se puede eliminar del análisis.

Ejemplo Cajero: Durante una transacción de retiro de dinero es necesario llevar un conteo de cuantos billetes están siendo enviados al cliente. Podría existir un atributo en Transacción que mantuviera esta información, la cual es sólo de interés interno, y no debiera ser listado como un atributo durante el análisis.

- Se deben evitar durante esta etapa los atributos derivados. Los atributos derivados se pueden incluir para mejorar la eficacia de la implementación, siendo esto un aspecto del diseño. Durante el análisis se deben omitir los atributos derivados a menos que se considere importante incluirlos. Si se incluyen, se deben identificar con notación especial.

2.2.2.3.3 Diagrama de Clases con Atributos

No es necesario listar todos los atributos. Se deben guardar los atributos más importantes. Se debe omitir atributos menores que posiblemente no afecten la mayoría de las operaciones.

El diagrama de clases para el sistema de cajeros, incluyendo los atributos (además de las asociaciones) de las clases, se muestra en la Figura 2.148.

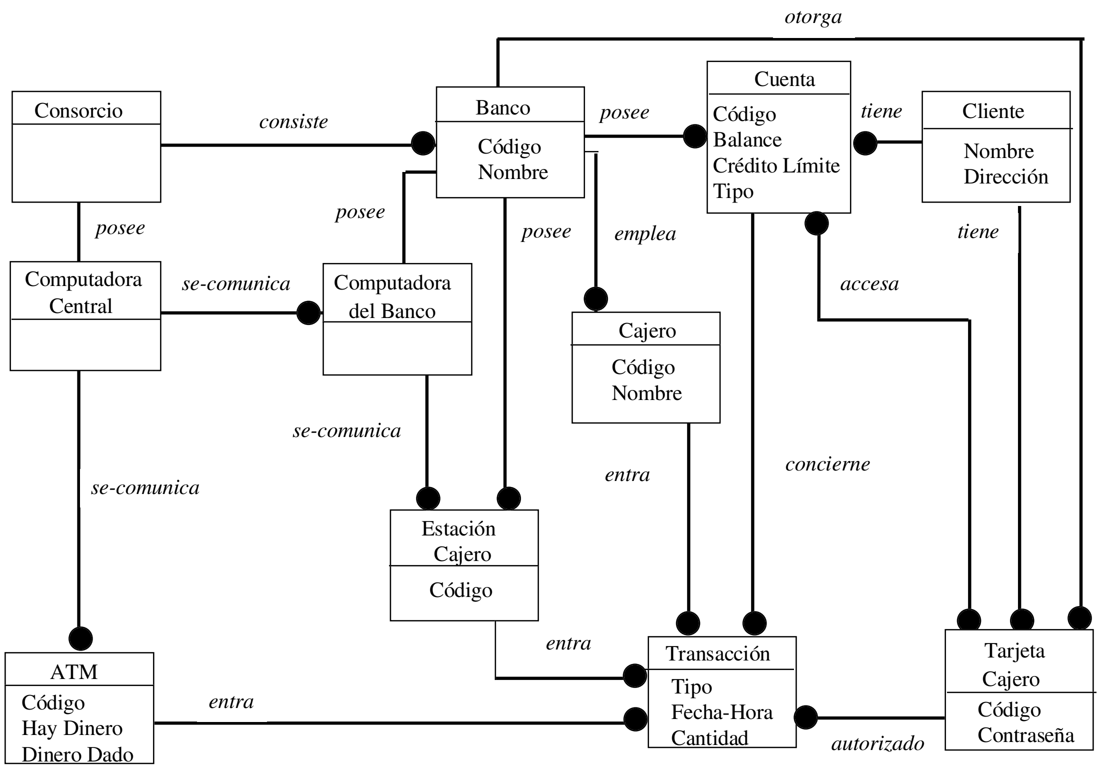

Figura 2.148. Diagrama de clases para el sistema de cajeros, incluyendo asociaciones y atributos.

2.2.2.3.4 Refinamiento de Clases

A continuación se puede hacer un refinamiento sobre el diagrama de clases con atributos. Opcionalmente se puede continuar con el análisis del modelo de objetos para posteriormente refinar los atributos, como será hecho con el ejemplo actual del *Sistema de Cajeros*.

2.2.2.4 Herencia

Se puede introducir jerarquías de herencia de la siguiente manera:

- herencia de abajo hacia arriba,
- herencia de arriba hacia abajo,
- herencia múltiple

2.2.2.4.1 Herencia de Abajo hacia Arriba

Se deben generalizar aspectos comunes de las clases existentes, creando superclases para compartir los atributos y operaciones comunes.

Ejemplo Cajero: *Estación de Entrada* puede generalizar a *Máquina de Cajero Automático* y *Estación de Cajero Humano*. Por otro lado *Computadora central* y *Computadora del banco* podrían ser generalizadas en *Computadora*, pero en este ejemplo tienen poco en común que se pueda compartir.

- Cuando una misma asociación aparece más de una vez en el diagrama, con el mismo nombre y significado, se debe generalizar la asociación a la superclase.

Ejemplo Cajero: Las *Cuentas conciernen a Transacción de Cajero Automático* y a *Transacción de Cajero Humano*, se pueden convertir en una sola asociación entre *Cuenta* y *Transacción*.

- La generalización de abajo hacia arriba es bastante aparente en la aplicación.

2.2.2.4.2 Herencia de Arriba hacia Abajo

Se deben refinar clases generales existentes en subclases especializadas.

Ejemplo Cajero: *Transacción de Cajero Automático* y *Transacción de Cajero Humano* pueden especializar a la clase *Transacción*.

- Las subclases enumeradas son una fuente muy frecuente para la especialización, donde muchas veces estas existen aunque no estén listadas.

Ejemplo Cajero: *Cuenta* puede ser refinada en *Cuenta de Cheques* y *Cuenta de Ahorro*. Esta distinción no es importante en el ejemplo del sistema de cajeros, y *Tipo* de *Cuenta* puede ser dejado como un atributo de *Cuenta*.

2.2.2.4.3 Herencia Múltiple

La herencia múltiple se puede utilizar para refinar el modelo. Por lo general se utiliza sólo si es importante, ya que aumenta la complejidad conceptual y la de implementación.

Ejemplo Cajero: Se podrían crear dos superclases, *Cajero Remoto* y *Cajero Local*. *Estación de Cajero Humano* entonces heredaría a la vez de *Estación de Entrada* y *Cajero Local*, y *Máquina de Cajero Automático* heredaría a la vez de *Estación de Entrada* y *Cajero Remoto*. De forma similar se puede aplicar herencia múltiple a la jerarquía de *Transacción*.

2.2.2.4.4 Diagrama de Clases con Herencia

El diagrama de clases para el sistema de cajero, incluyendo herencia, se muestra en la Figura 2.149.

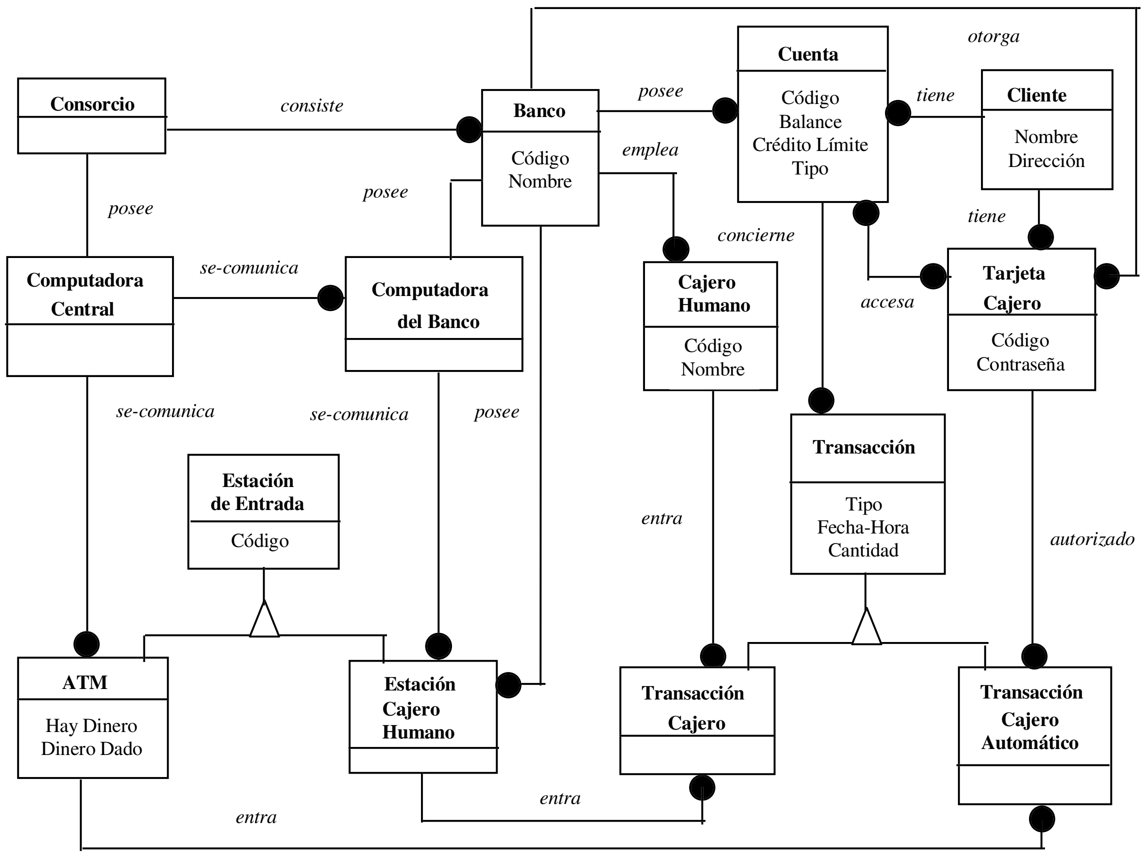

Figura 2.149. Diagrama de clases para el sistema de cajero, incluyendo herencia.

2.2.3 Verificación del Modelo

Una vez terminada la elaboración del modelo inicial, corresponde realizar su verificación. Se deben verificar las asociaciones, analizando en el diagrama los caminos de acceso entre clases, para ver si los resultados son los deseados.

- Se deben analizar posibles preguntas y ver si el modelo las responde, y si no, el modelo debe ser modificado.

Ejemplo Cajero: Una *Tarjeta de Cajero* no identifica de forma única una *Cuenta* de cheque o ahorro, y el usuario debe elegir de alguna forma entre las dos. Cada *Tarjeta de Cajero* debe accesar no más de una cuenta de cheques o ahorro. Esto puede que sea razonable pero limita al sistema. La alternativa es pedir a los clientes que recuerden sus números de *Cuenta*, pero si se accede a cuentas únicas es imposibles hacer transferencias entre cuentas.

- Si hacen falta caminos de acceso para operaciones, se deben añadir nuevas asociaciones para que puedan ser llamadas.

- Si hacen falta operaciones que atraviesan la asociación, quizás la información no es necesaria.

- Todas las consultas debe poder ser contestadas.

- Si existe información redundante en la asociación, se deben eliminar las asociaciones que no añadan nueva información, o marcarlas como derivadas.

- Si existen nombres de rol demasiado imprecisos o muy específicos para sus clases, se debe mover la asociación hacia arriba o abajo de la herencia de clases.

2.2.4 Iteración del Modelo

El modelo de objeto es raramente correcto la primera vez, requiriendo múltiples iteraciones para lograr completarse. Todo el proceso de desarrollo de software es un proceso de iteración, donde diferentes partes del modelo son hechas durante diferentes etapas.

- Si se encuentra una deficiencia se debe ir a una etapa anterior para corregirlo.

- La experiencia hará que se pueda variar el orden de estos pasos, pero inicialmente se deben seguirlos en su orden.

- Algunos refinamientos se pueden hacer solamente después de haber completados los modelos dinámico y funcional.

Ejemplo Cajero: La distinción entre *Banco* y *Computadora del Banco,* y entre *Consorcio* y *Computadora Central* no parecen afectar el análisis. El hecho de que las comunicaciones sean procesadas por computadoras es en realidad un concepto de implementación. Se puede unificar *Computadora del Banco* con *Banco* y *Computadora Central* con *Consorcio*.

El diagrama de clases para el sistema de cajero con un refinado final se muestra en la Figura 2.150.

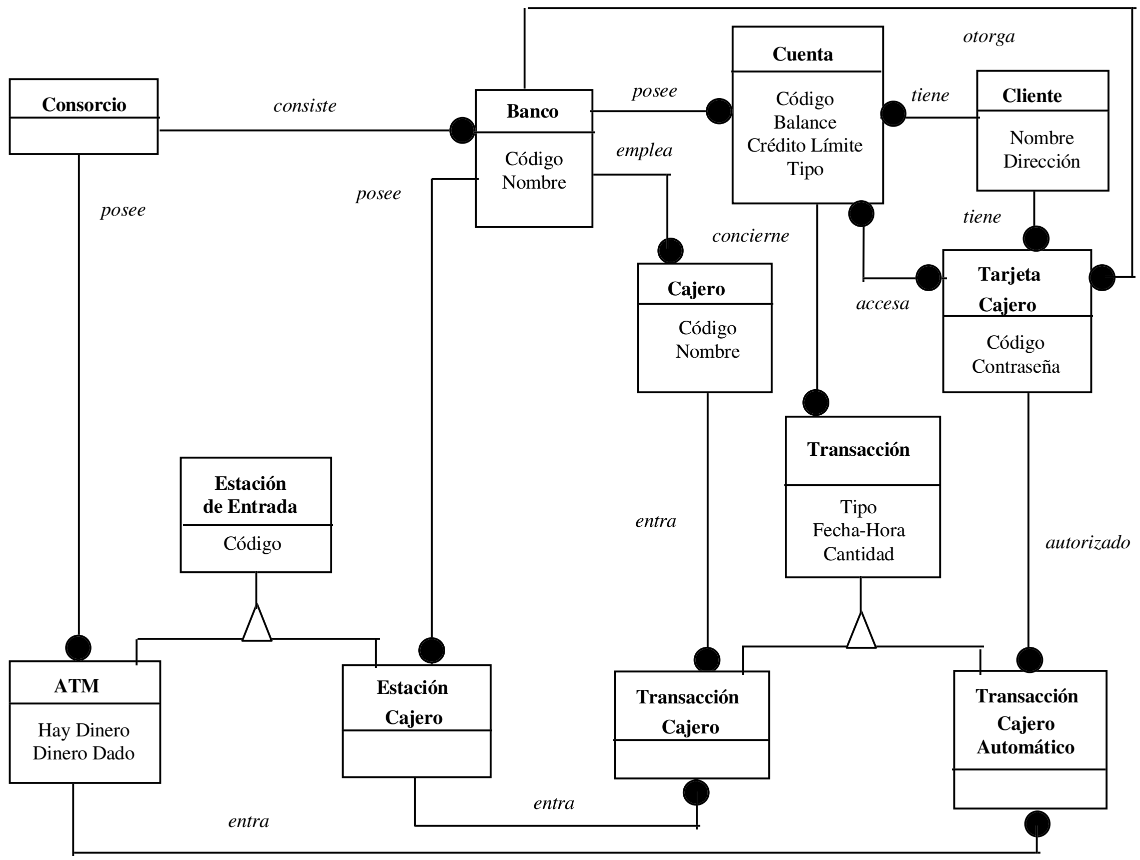

Figura 2.150. Diagrama final de clases para el sistema de cajeros.

2.2.5 Diccionario de Datos

Los nombres de las clases pueden tener muchas interpretaciones, por lo cual se debe preparar un *diccionario de datos* describiendo las diferentes entidades del modelo de forma precisa.

- Como parte del diccionario de datos se debe escribir un párrafo describiendo cada clase de objeto, sus asociaciones, los atributos y las operaciones.

- Se debe definir un formato general para definir las clases.

- Se debe actualizar el diccionario de datos durante las diferentes etapas del análisis del modelo de objetos.

Ejemplo Cajero: el siguiente es el Diccionario de Datos para el *Sistema de Cajero* (organizado alfabeticamente):

Banco: Institución financiera que tiene cuentas de clientes y provee tarjetas de cajero automático autorizando acceso a cuentas sobre la red de los ATM.

Cajero Humano: Empleado del banco que está autorizado a entrar transacciones a estaciones de cajero, aceptar, dar dinero efectivo, y dar cheques a clientes. Las transacciones, efectivo, y cheques, manejados por cada cajero deben ser registrados.

Cliente: El poseedor de una o más cuentas en el banco. El cliente puede consistir de una o más personas, o corporaciones. La misma persona que posee una cuenta en diferentes bancos se puede considerar un cliente diferente.

Consorcio: Organización de bancos que comisiona y opera la red de los ATM. La red sólo maneja transacciones para bancos del consorcio.

Cuenta: Una cuenta en un banco contra la cual las transacciones pueden ser aplicadas. Las cuentas pueden ser de varios tipos, incluyendo cuentas de cheques o ahorro. El cliente puede tener más de una cuenta.

Estación de Entrada: Estación general de cajeros comunes a la estación de cajero humano y máquina de cajero automático.

Estación de Cajero Humano: Estación del banco en la cual los cajeros entran transacciones para los clientes. Los cajeros dan y aceptan dinero efectivo y cheques, e imprimen recibos. La estación de cajero se comunica con la computadora del banco para validar y procesar las transacciones.

Máquina de Cajero Automático (ATM): Máquina de cajero automático que permite al cliente entrar sus propias transacciones usando tarjetas de cajero automático como identificación. El ATM interacciona con el cliente durante la transacción, manda la información a la computadora central para validación y procesamiento, y deja retirar dinero al cliente. El ATM no opera independiente de la red.

Tarjeta de Cajero Automático: Una tarjeta asignada a un cliente del banco que autoriza el acceso a las cuentas usando la máquina ATM. Cada tarjeta contiene un código de banco y un número de tarjeta, posiblemente certificada de acuerdo a leyes nacionales de tarjetas de crédito y efectivo. El código del banco identifica claramente al banco dentro del consorcio. El número de tarjeta determina la cuenta que la tarjeta puede accesar. La tarjeta no accesa necesariamente todas las cuentas del cliente. Cada tarjeta pertenece a un solo cliente, pero múltiples copias pueden existir, y la posibilidad de uso simultáneo de la misma tarjeta en diferentes máquinas debe ser considerado.

Transacción: Una operación en una cuenta de un solo cliente. Las transacciones incluyen retiro de dinero efectivo y depósitos de efectivo o cheques. Las diferentes operaciones tienen que estar balanceadas apropiadamente. Transacción común a la transacción de cajero humano y de máquina de cajero automático.

Transacción de Cajero Automático: Transacción que requiere de una tarjeta de cajero automático para accesar la máquina de cajero humano.

Transacción de Cajero Humano: Transacción que se hace en la estación de cajero humano del banco por medio del cajero humano.

2.2.6 Agrupación en Módulos

El modelo de objetos puede hacerse muy complejo, para lo cual es necesario separarlo en *Módulos* para controlar su complejidad. El modelo es una colección de módulos, donde cada módulo es una agrupación lógica de clases, asociaciones y generalizaciones. Una *Página* es una agrupación física de clases, asociaciones y generalizaciones.

Ejemplo Cajero: El sistema de cajero no es lo suficientemente grande para necesitar una separación en módulos, pero podría servir como base para modelos más detallados.

• acceso: *cajeros*, *estación de entrada*, *estación de cajero*, *máquina de cajero automático*
• cuenta: *cuenta*, *tarjeta de cajero*, *tarjeta autorización*, *cliente*, *transacción*, *actualización*, *transacción cajero humano*, *transacción cajero automático*
• banco: *consorcio*, *banco*

Cada módulo puede añadir detalles:
• El módulo de *cuenta* puede añadir transacciones sobre clientes, pago de intereses, cargos
• El módulo de *banco* puede contener información sobre sucursales, direcciones y asignación de costos.