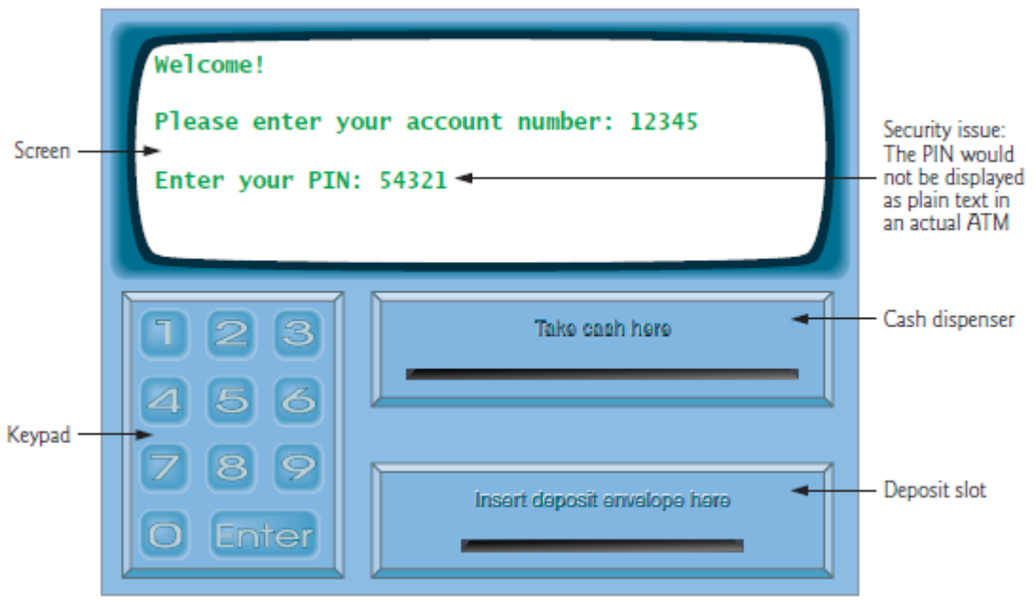
UML: ATM (CASE STUDY)

Sebuah bank lokal bermaksud untuk menginstal mesin teller otomatis baru (ATM) untuk memungkinkan pengguna (yaitu, Nasabah bank) untuk melakukan transaksi keuangan dasar (Gambar 1.1). Setiap user dapat memiliki hanya satu Account di bank. Pengguna ATM harus dapat melihat Saldo rekening mereka, Menarik uang kas (yaitu, mengambil uang dari rekening) dan Menyimpan dana (yaitu, menempatkan uang ke account).

User Interface dari mesin teller otomatis berisi:

- Screen(layar) yang menampilkan pesan kepada pengguna
- Keypad yang menerima input numerik dari pengguna
- Cash Dispenser (dispenser kas) yang membagi-bagikan uang kas kepada pengguna
- Deposit Slot (slot penyimpanan) yang menerima amplop setoran dari pengguna.

Dispenser kas setiap hari diisi dengan 500 pecahan $ 20.
[Catatan: Karena keterbatasan ruang lingkup studi kasus ini, unsur-unsur tertentu dari ATM yang dijelaskan di sini tidak akurat meniru ATM sebenarnya. Sebagai contoh, ATM nyata biasanya berisi sebuah perangkat yang membaca nomor rekening pengguna dari kartu ATM, sedangkan ATM ini meminta pengguna untuk mengetik nomor rekening pada keypad. Sebuah ATM nyata juga biasanya mencetak tanda terima pada akhir sesi, tetapi semua output dari ATM ini muncul di layar.



Gambar 1.1. User Interface ATM

- Bank menginginkan Anda untuk mengembangkan perangkat lunak untuk melakukan transaksi keuangan dimulai oleh nasabah bank melalui ATM.
- Bank akan mengintegrasikan perangkat lunak dengan hardware ATM di lain waktu.

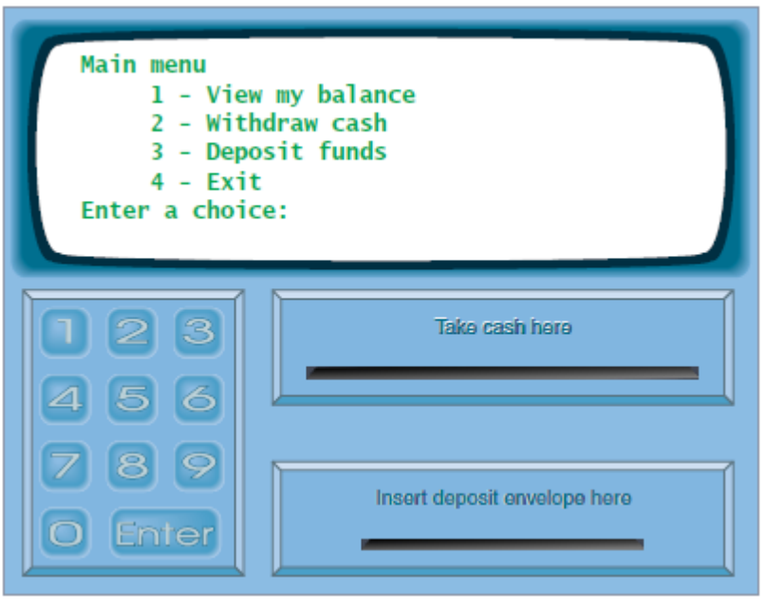
- Perangkat lunak ini harus merangkum fungsionalitas dari perangkat keras (misalnya, dispenser kas, slot penyimpanan) dalam komponen perangkat lunak, namun tidak perlu mempedulikan bagaimana perangkat ini melakukan tugasnya.
- Perangkat keras ATM belum dikembangkan, jadi daripada menulis perangkat lunak Anda untuk berjalan di ATM, Anda harus mengembangkan versi pertama yang dijalankan pada PC.
- Versi ini harus menggunakan monitor komputer untuk mensimulasikan layar ATM, dan keyboard komputer untuk mensimulasikan keypad ATM.

Sesi ATM terdiri dari:

1. Otentikasi Pengguna (yaitu, membuktikan identitas pengguna) berdasarkan nomor rekening dan nomor identifikasi pribadi (PIN), yang diikuti dengan Menciptakan dan Melaksanakan Transaksi keuangan. Untuk mengotentikasi pengguna dan melakukan transaksi,
2. ATM harus berinteraksi dengan Database Infomasi rekening milik bank (yaitu, koleksi terorganisir dari data yang tersimpan pada komputer). Untuk setiap rekening bank, database menyimpan nomor rekening, PIN dan saldo yang menunjukkan jumlah uang di rekening. [Catatan: Kami berasumsi bahwa bank berencana untuk membangun hanya satu ATM, jadi kita tidak perlu khawatir tentang beberapa ATM mengakses database ini pada waktu yang sama. Selain itu, kami berasumsi bahwa bank tidak membuat perubahan apapun informasi dalam database sementara pengguna mengakses ATM. Kami membuat asumsi penyederhanaan, bagaimanapun, bahwa bank mempercayai ATM untuk mengakses dan memanipulasi informasi dalam database tanpa langkah-langkah keamanan yang signifikan.]

Setelah pengguna berjalan mendekati ATM (dengan asumsi tidak ada orang yang sedang menggunakannya), Pengguna harus mengalami urutan peristiwa berikut (ditunjukkan pada Gambar 1.1.):

1. Layar Selamat Datang! dan meminta pengguna untuk memasukkan nomor rekening.
2. Pengguna memasukkan nomor rekening lima digit menggunakan tombol.
3. Layar meminta pengguna untuk memasukkan PIN (nomor identifikasi pribadi) terkait dengan nomor rekening tertentu.
4. Pengguna memasukkan PIN lima digit menggunakan keypad
5. Jika pengguna memasukkan nomor rekening valid dan PIN yang benar untuk account tersebut, layar menampilkan menu utama (Gambar 1.2). Jika pengguna memasukkan account yang tidak valid nomor atau PIN yang salah, layar menampilkan pesan yang tepat, maka ATM kembali ke Langkah 1 untuk memulai kembali proses otentikasi



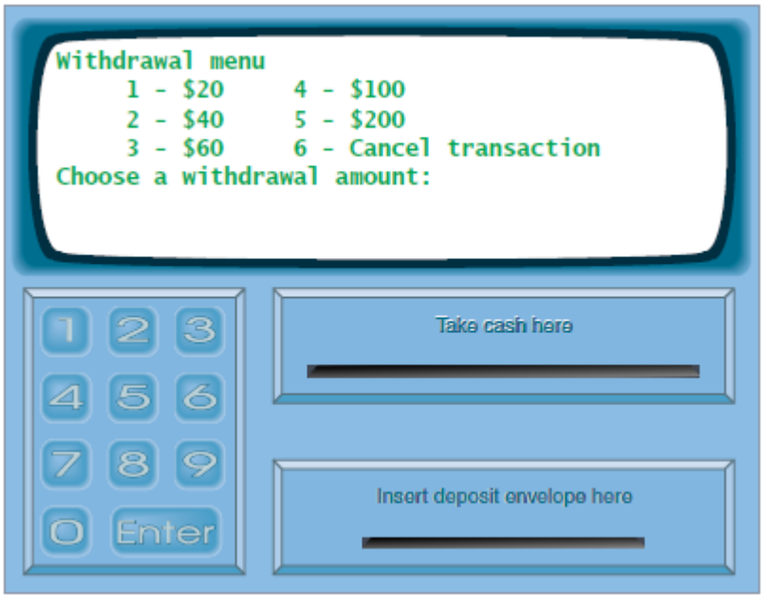
Gambar 1.2. Menu Utama ATM

Setelah ATM mengotentikasi pengguna, menu utama (Gambar 1.2) harus berisi option nomor untuk masing-masing tiga jenis transaksi: Informasi Saldo (opsi 1), Penarikan (opsi 2) dan Penyimpanan Dana (opsi 3). Hal ini juga harus berisi opsi untuk memungkinkan pengguna untuk keluar dari sistem (option 4). Pengguna kemudian memilih baik untuk melakukan transaksi (dengan memasukkan 1, 2 atau 3) atau keluar dari sistem (dengan memasukkan 4).

Jika pengguna memasukkan 1 untuk melihat Informasi Saldo, layar menampilkan Saldo rekening pengguna. Untuk melakukannya, ATM harus mengambil nilai Saldo dari database milik bank. Langkah-langkah berikut menjelaskan apa yang terjadi ketika pengguna memasukkan 2 untuk melakukan penarikan:

1. Layar menampilkan menu (Gambar 1.3) mengandung jumlah penarikan standar: $ 20 (opsi 1), $ 40 (opsi 2), $ 60 (opsi 3), $ 100 (option 4) dan $ 200 (option 5). Menu ini juga berisi opsi untuk memungkinkan pengguna untuk membatalkan transaksi (opsi 6).
2. Pengguna memasukkan pilihan menu menggunakan tombol.
3. Jika jumlah penarikan yang dipilih lebih besar dari saldo account pengguna, layar menampilkan pesan yang menyatakan ini dan memberitahu pengguna untuk memilih jumlah yang lebih kecil. ATM kemudian kembali ke Langkah 1. Jika jumlah penarikan yang dipilih adalah kurang dari atau sama dengan saldo rekening pengguna (yaitu, jumlah yang dapat diterima), ATM ke Langkah 4. Jika pengguna memilih untuk membatalkan transaksi (option 6), ATM menampilkan menu utama dan menunggu input pengguna.
4. Jika dispenser kas mengandung cukup uang kas, ATM hasil ke Langkah 5. Jika tidak, layar menampilkan pesan yang menunjukkan masalah dan memberitahu pengguna untuk memilih jumlah penarikan yang lebih kecil. ATM kemudian kembali ke Langkah 1.
5. ATM mendebit jumlah penarikan dari account pengguna di bank Database (yaitu, mengurangi jumlah penarikan dari saldo rekening Pengguna).
6. ATM membagi-bagikan jumlah yang diinginkan uang kepada Pengguna.

7. Layar menampilkan pesan yang mengingatkan pengguna untuk mengambil uang.



Gambar 1.3. Menu Penarikan ATM

Langkah-langkah berikut menjelaskan tindakan yang terjadi ketika pengguna memasukkan 3 (bila melihat menu utama Gambar . 1.2) untuk melakukan Penyimpanan Dana:

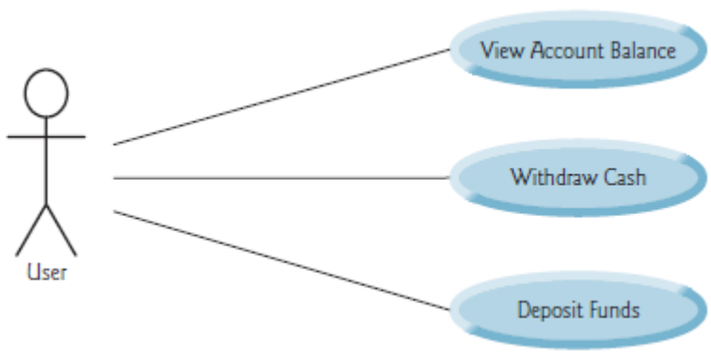
1. Layar meminta pengguna untuk memasukkan jumlah deposit atau masukkan 0 (nol) untuk membatalkan.
2. Pengguna memasukkan jumlah deposit atau 0 menggunakan tombol. [Catatan: Tombol tidak mengandung titik desimal atau tanda dolar , sehingga user tidak bisa mengetik nilai uang sebenarnya (misalnya , $ 27,25). sebaliknya , pengguna harus memasukkan jumlah deposit sebagai angka sen (mis. , 2725) . ATM kemudian membagi angka ini dengan 100 untuk mendapatkan angka yang mewakili jumlah uang yang dimaksudkan (misalnya, 2725 ÷ 100 = 27,25) .]
3. Jika pengguna menentukan Jumlah Penyimpanan Dana, ATM menuju ke langkah 4 . Jika Pengguna memilih untuk membatalkan transaksi (dengan memasukkan 0) , ATM menampilkan menu utama dan menunggu input Pengguna.
4. Layar menampilkan pesan yang mengatakan pengguna untuk memasukkan Amplop Penyimpanan
5. Jika Slot Penyimpanan menerima amplop penyimpanan dalam waktu dua menit , ATM melakukan Kredit sejumlah Nilai Penyimpanan ke dalam rekening Pengguna dalam database bank (yaitu, menambahkan jumlah penyimpanan ke saldo Rekening milik pengguna). [Catatan : Uang ini tidak segera tersedia untuk penarikan. Bank harus terlebih dahulu secara fisik memverifikasi jumlah uang kas dalam amplop deposito, dan cek dalam amplop harus jelas (yaitu, uang harus ditransfer dari rekening cek penulis untuk cek penerima rekening). Ketika salah satu dari peristiwa-peristiwa ini terjadi , bank akan memutakhirkan data saldo pengguna yang tersimpan dalam database. Hal ini terjadi secara independen dari Sistem ATM .] Jika slot penyimpanano tidak menerima amplop deposito dalam periode tertentu, layar menampilkan pesan bahwa sistem telah membatalkan transaksi karena tidak aktif. ATM kemudian menampilkan menu utama dan menunggu input pengguna. Setelah sistem berhasil mengeksekusi transaksi,

harus kembali ke menu utama sehingga pengguna dapat melakukan transaksi tambahan. Jika pengguna keluar dari sistem, layar harus menampilkan pesan ucapan terima kasih , kemudian menampilkan pesan selamat datang bagi pengguna berikutnya .

USE CASE DIAGRAM

Kita sekarang memperkenalkan yang pertama dari beberapa diagram UML dalam studi kasus . Kami membuat sebuah Use Case Diagram untuk memodelkan interaksi antara sebuah *Client* dari sistem (dalam studi kasus ini, Nasabah bank) dan Use Case-nya. Tujuannya adalah untuk menunjukkan jenis interaksi pengguna dengan sistem tanpa memberikan rincian – hal ini diberikan dalam diagram UML lainnya (yang disajikan di seluruh studi kasus ini). Diagram use case sering disertai oleh teks informal yang memberikan tambahan rincian seperti teks yang muncul dalam dokumen persyaratan (Requirements Document). Use Case Diagram dihasilkan selama tahap analisis siklus hidup perangkat lunak . Dalam sistem yang lebih besar, Use Case Diagram adalah alat yang sangat diperlukan yang membantu desainer sistem tetap fokus pada pemuasan kebutuhan pengguna.

Gambar 1.4 menunjukkan Use Case Diagram untuk sistem ATM. Figur batang mewakili seorang aktor, yang mendefinisikan peran dari entitas eksternal - misalnya orang atau sistem lain - saat berinteraksi dengan sistem. Untuk ATM ini, aktor adalah Pengguna yang dapat Melihat Saldo rekening , Penarikan Tunai dan Penyimpanan Dana dari ATM. Pengguna bukanlah orang yang sebenarnya , melainkan terdiri dari peran yang nyata dari seseorang - ketika memainkan peran dari Pengguna – dapat berperan sambil berinteraksi dengan ATM . Sebuah Use Case Diagram dapat mencakup beberapa aktor.



Gambar 1.4. Use Case Diagram untuk ATM dari Perspektif Pengguna

Sebagai contoh, penggunaan Use Case Diagram sistem ATM bank sebenarnya juga mungkin mencakup seorang aktor bernama Administrator yang melakukan Pengisian Ulang uang kas Dispenser ATM setiap hari.

IDENTIFIKASI CLASS DI DALAM SISTEM

Nouns and noun phrases in the ATM requirements document			
bank	money / funds	account number	ATM
screen	PIN	user	keypad
bank database	customer	cash dispenser	balance inquiry
transaction	$20 bill / cash	withdrawal	account
deposit slot	deposit	balance	deposit envelope

Gambar 1.5. Kata Benda dan Frasa di dalam Dokumen Kebutuhan ATM

Kita membuat class hanya untuk kata benda dan frase kata benda yang memiliki signifikansi dalam Sistem ATM. Kita tidak memodelkan "bank" sebagai sebuah Class, karena bank bukan merupakan bagian dari sistem ATM. Bank ATM hanya ingin kita untuk membangun ATM. "Pelanggan" dan "User" juga mewakili entitas luar - mereka penting karena mereka berinteraksi dengan sistem ATM kita, tetapi kita tidak perlu model mereka sebagai Class dalam perangkat lunak ATM. Ingatlah bahwa kita memodelkan pengguna ATM (yaitu, Nasabah bank) sebagai aktor dalam diagram use case Gambar. 1.4.

Kita tidak memodelkan "20 dolar" atau "Amplop Deposit" sebagai Class. Mereka adalah bagian dari object fisik di dunia nyata, tapi mereka bukan bagian dari apa yang sedang dibuat otomatis. Kita cukup dapat mewakili kehadiran tagihan dalam sistem menggunakan atribut class yang memodelkan dispenser uang kas. (Kita menetapkan atribut untuk class Sistem ATM di Bagian 1.4). Sebagai contoh, dispenser kas memelihara hitungan jumlah uang dolar di dalam ATM. Dokumen Persyaratan (Requirements Document) tidak mengatakan apa-apa tentang apa yang harus dilakukan oleh sistem terhadap amplop deposito setelah menerima mereka. Kita dapat berasumsi bahwa dengan hanya mengenali penerimaan amplop - operasi yang dilakukan oleh class yang memodelkan slot penyimpanan uang - sudah cukup untuk mewakili kehadiran sebuah amplop di dalam sistem.

Dalam sistem ATM kita yang disederhanakan, mewakili sejumlah "Uang”, meliputi “Saldo” rekening sebagai atribut class tampaknya paling tepat. Demikian juga, kata benda "Nomor Rekening" dan "PIN" mewakili potongan besar informasi di dalam Sistem ATM. Mereka adalah atribut penting dari sebuah rekening bank. Mereka, bagaimanapun, tidak memperlihatkan Perilaku. Dengan demikian , kita dapat paling tepat model mereka sebagai atribut dari class Rekening. Meskipun dokumen persyaratan sering menggambarkan "transaksi" dalam pengertian umum , kita tidak memodelkan gagasan luas transaksi keuangan saat ini. Sebaliknya , kita memodelkan tiga jenis transaksi (yaitu , "Cek Saldo” , “Penarikan”, dan “Penyimpanan”) sebagai tiga class individu.

Ketiga class ini memiliki atribut khusus yang dibutuhkan untuk melaksanakan transaksi yang mereka wakili . Sebagai contoh, Penarikan perlu mengetahui jumlah penarikan. Sebuah Cek Saldo, bagaimanapun, tidak memerlukan data tambahan selain Nomor Rekening. Lebih lanjut, tiga class transaksi menunjukkan perilaku unik. Penarikan

mencakup Pengeluaran Kas kepada Pengguna, sedangkan Penyimpanan Dana melibatkan menerima amplop setoran dari Pengguna. Dalam Bagian 13.3 , kita akan melihat fitur-fitur umum dari semua transaksi menjadi class “transaksi” umum menggunakan konsep berorientasi object Pewarisan .

Kita menentukan class untuk sistem berdasarkan pada kata benda yang tersisa dan kata benda frase dari Gambar. 1.5. Masing-masing mengacu pada satu atau lebih hal berikut:

- ATM
- screen
- keypad
- cashdispenser
- deposit slot
- account
- bank database
- balance inquiry
- withdrawal
- deposit

Elemen dari daftar di atas yang nantinya akan kita buat menjadi Class yang diperlukan oleh Sistem:

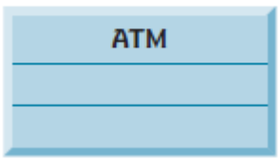
- ATM
- Screen
- Keypad
- CashDispenser
- DepositSlot
- Account
- BankDatabase
- BalanceInquiry
- Withdrawal
- Deposit

Kita membangun sistem menggunakan Class di atas sebagai elemen dasar. Sebelum memulai membangun sistem, namun, kita harus memperoleh pemahaman lebih baik tentang keterkaitan antar class.

PEMODELAN CLASS

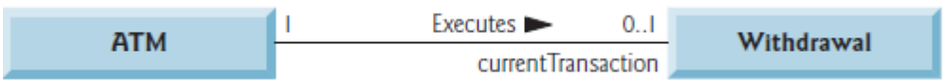
UML memungkinkan kita untuk memodelkan, menggunakan Class Diagram, class di dalam Sistem ATM dan hubungan diantara class. Gambar 1.6 mewakili class ATM. Setiap class dimodelkan sebagai sebuah segi empat dengan tiga ruang. Bagian paling atas berisi nama class diletakkan di tengah dengan cetak tebal. Bagian tengah berisi atribut dari class. Bagian paling bawah berisi operasi dari class.

Di dalam gambar 1.6, bagian tengah dan bawah masih kosong karena belum ditentukan.



Gambar 1.6. Mewakili Sebuah Class Menggunakan Sebuah Class Diagram

Class diagram juga menunjukkan hubungan antara Class dari sistem. Gambar 1.7 menunjukkan bagaimana class ATM dan Withdrawal saling terhubung. Untuk sekarang, untuk kesederhanaan, kita pilih untuk hanya memodelkan subset dari class ini.



Gambar 1.7. Class Diagram yang Menunjukkan Sebuah Asosiasi Antar Class

Perhatikan bahwa persegi panjang yang mewakili class tidak terbagi atas ruang. UML mengijinkan penghilangan atribut dan operasi milik class untuk membuat diagram yang lebih mudah untuk dibaca secara layak. Digram tersebut disebut dengan Elided Diagram – diagram yang beberapa informasi di dalamnya seperti isi dari ruang ke-2 dan ke-3 tidak dimodelkan.

Dalam Gambar. 1.7, garis padat yang menghubungkan dua class merupakan sebuah asosiasi – sebuah relasi antar class. Angka-angka di dekat setiap akhir baris adalah Nilai Multiplicity, yang menunjukkan berapa banyak object dari setiap class berpartisipasi dalam asosiasi. Dalam contoh ini, jika kita ikuti garis dari kiri ke kanan akan menunjukkan bahwa, pada saat tertentu, setiap object ATM berpartisipasi di dalam sebuah asosiasi dengan baik nol atau satu object Withdrawal – nol, jika User tidak sedang melakukan transaksi atau meminta jenis transaksi lain, dan bernilai satu jika user meminta transaksi Withdrawal. UML dapat memodelkan banyak jenis multiplicity. Lihat gambar 1.8.

Symbol	Meaning
0	None
1	One
m	An integer value
0..1	Zero or one
m, *n*	*m* or *n*
m..*n*	At least *m*, but not more than *n*
*	Any nonnegative integer (zero or more)
0..*	Zero or more (identical to *)
1..*	One or more

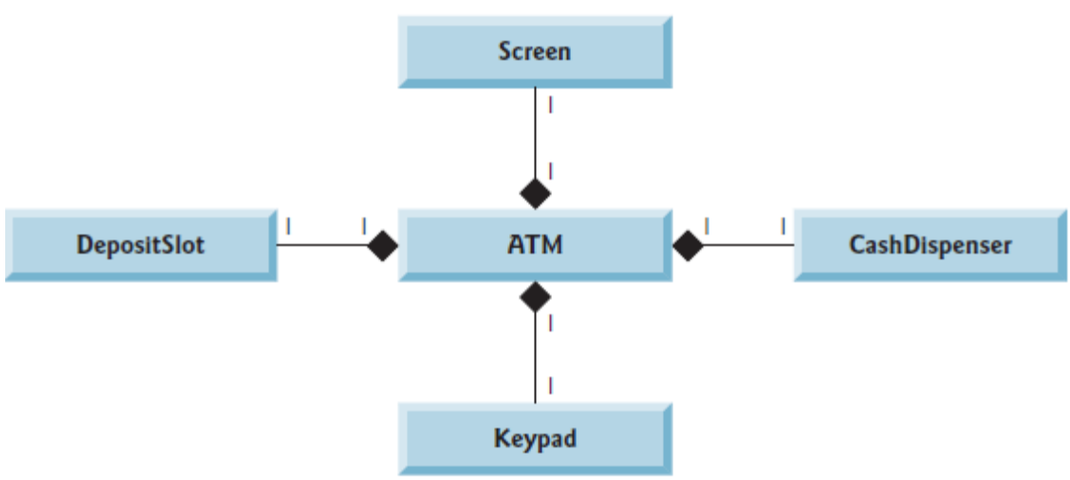
Gambar 1.8. Jenis Multiplicity

Sebuah Asosiasi dapat diberi nama. Sebagai contoh, kata “Executes” di atas garis yang menghubungkan class ATM dan Withdrawal pada Gambar 1.7 menandakan nama dari asosiasi tersebut. Bagian dari diagram ini dapat dibaca “sebuah object dari class ATM executes nol atau satu object dari class Withdrawal”. Nama asosiasi dibaca sesuai arah anak panah.

Kata currentTransaction pada sisi Withdrawal dari tanda panah di dalam Gambar 1.7 adalah sebuah nama peran (role name), menandakan peran yang dijalankan oleh object Withdrawal di dalam relasi dengan ATM. Nama peran (role name) menambahkan arti kepada sebuah asosiasi antara class dengan mengidentifikasi peran sebuah class di dalam konteks sebuah asosiasi. Sebuah class dapat menjalankan beberapa peran di dalam sistem yang sama. Sebagai contoh, di dalam sebuah sistem personalia sekolah, seseorang dapat berperan sebagai “Pengajar” ketika berelasi dengan Murid. Orang yang sama dapat berperan sebagai “Kolega” ketika ambil bagian di dalam sebuah asosiasi dengan Pengajar lainnya, dan “Pelatih” jika sedang mengajar atlit pelajar.

Di dalam Gambar 1.7, nama peran currentTransaction menandakan bahwa object Withdrawal yang ikut serta di dalam asosiasi Executes dengan sebuah object dari class ATM mewakili transaksi yang sedang diproses oleh ATM. Di dalam konteks lain, sebuah object Withdrawal dapat mengambil peran lain (misalnya “previous transaction”). Perhatikan bahwa kita tidak menentukan sebuah nama peran untuk sisi ATM dari asosiasi Executes. Nama Peran di dalam class diagram seringkali diabaikan ketika makna dari asosiasi telah jelas tanpanya.

Selain menunjukkan hubungan sederhana, asosiasi dapat menentukan lebih kompleks hubungan, seperti object dari satu class yang terdiri dari benda-benda dari class lain. bayangkan ATM dunia nyata. Bagian apa sajakah yang disatukan bersama-sama oleh produsen agar sebuah ATM bekerja? Dokumen persyaratan memberitahu kita bahwa ATM adalah terdiri dari sebuah layar, sebuah keypad, dispenser kas dan slot penyimpanan.



Gambar 1.9. Class Diagram Menunjukkan Relasi Komposisi

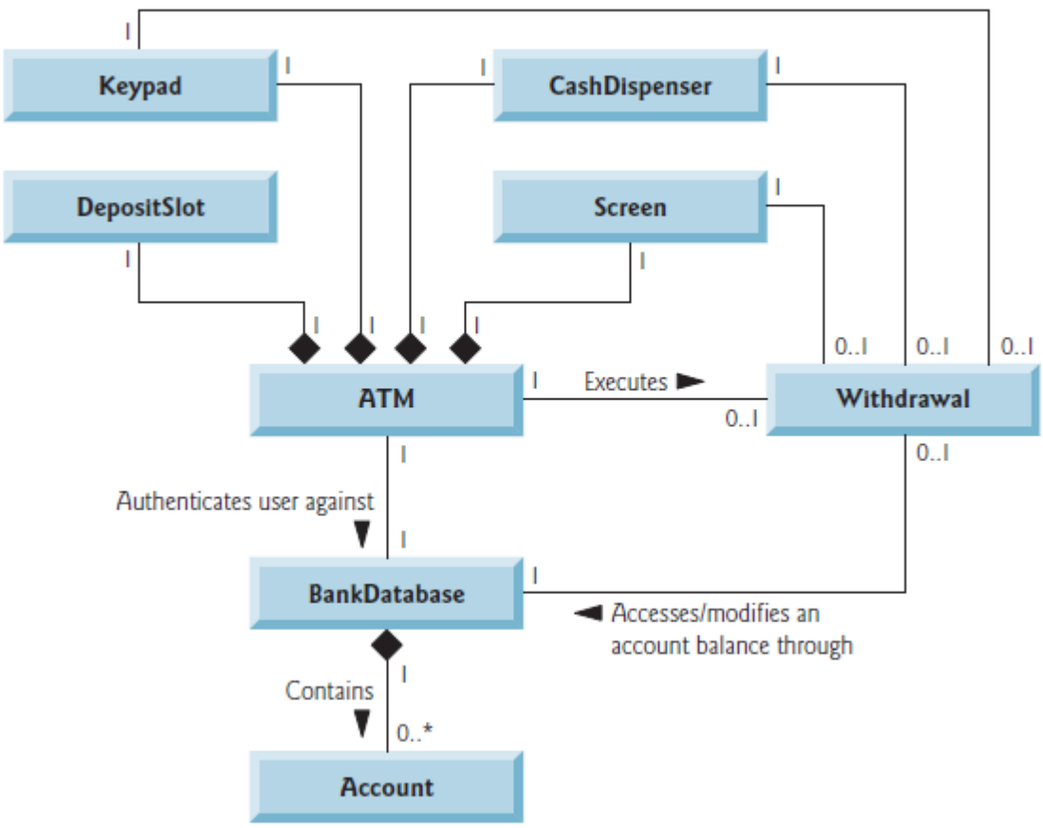
Dalam Gambar 1.9, berlian penuh yang melekat pada garis asosiasi class ATM menunjukkan ATM yang memiliki hubungan dengan komposisi class Screen, Keypad, CashDispenser dan DepositSlot. Komposisi menyiratkan hubungan seluruhnya / sebagian. Class yang memiliki simbol komposisi (berlian penuh) pada akhir dari garis asosiasi adalah keseluruhan (dalam hal ini, ATM), dan class di ujung lain dari garis asosiasi adalah bagian - dalam hal ini, Screen , Keypad, CashDispenser dan DepositSlot. Komposisi pada Gambar 1.9 menunjukkan bahwa sebuah object dari class ATM class terbentuk dari satu object dari class Screen, salah satu object dari class CashDispenser, salah satu object dari class Keypad dan satu object dari class DepositSlot. ATM memiliki sebuah layar, sebuah keypad, sebuah dispenser kas dan sebuah slot penyimpanan.

Menurut spesifikasi UML (www.omg.org/technology/documents/ formal/uml.htm), relasi hubungan komposisi memiliki sifat sebagai berikut:

1. Hanya satu class dalam hubungan yang dapat mewakili keseluruhan (yaitu, berlian dapat ditempatkan hanya pada satu ujung garis asosiasi). Sebagai contoh, baik layar adalah bagian dari ATM atau ATM merupakan bagian dari layar, tapi layar dan ATM tidak bisa keduanya mewakili keseluruhan dalam relasi.
2. Bagian dalam relasi komposisi hanya ada selama keseluruhan ada, dan keseluruhan bertanggung jawab untuk penciptaan dan penghancuran bagian-bagiannya. Sebagai contoh, tindakan membangun sebuah ATM meliputi pembuatan bagian-bagiannya. Juga, jika ATM dihancurkan, layar, keypad, dispenser kas dan slot penyimpanan juga hancur.
3. Sebuah bagian mungkin hanya dimiliki oleh sebuah keseluruhan pada suatu waktu, meskipun dapat dihilangkan dan dilekatkan kepada sebuah keseluruhan lain, yang kemudian bertanggung jawab untuk bagian tersebut.

Berlian penuh dalam class diagram menunjukkan hubungan komposisi yang memenuhi sifat ini. Jika sebuah relasi has-a tidak memenuhi satu atau lebih kriteria ini, UML menetapkan bahwa berlian hampa dilekatkan pada ujung garis asosiasi untuk menunjukkan agregasi - bentuk yang lebih lemah dari komposisi. Sebagai contoh, sebuah komputer pribadi dan monitor komputer berpartisipasi dalam relasi agregasi - komputer memiliki monitor, tetapi keduanya dapat ada secara mandiri, dan monitor yang sama dapat dihubungkan ke beberapa komputer sekaligus, sehingga melanggar sifat kedua dan ketiga komposisi.

Gambar 1.10 menunjukkan class diagram untuk sistem ATM. Diagram memodelkan sebagian besar class yang kita telah identifikasi, serta asosiasi diantara mereka kita dapat simpulkan dari dokumen persyaratan. Class BalanceInquiry dan Deposit berpartisipasi dalam asosiasi yang sama dengan class Withdrawal, jadi kita telah memilih untuk menghilangkan mereka dari diagram ini untuk mejaganya tetap sederhana.



Gambar 1.10. Class Diagram Untuk Model Sistem ATM

Gambar 1.10 menyajikan model grafis dari struktur sistem ATM. Ini termasuk class BankDatabase dan Account, dan beberapa asosiasi yang ada baik pada Gambar 1.7 atau Gambar 1.9. Hal ini menunjukkan bahwa class ATM memiliki relasi one-to-one dengan class BankDatabase - satu object ATM mengotentikasi user terhadap satu object BankDatabase .

Dalam Gambar 1.10 , kita juga memodelkan fakta bahwa database milik bank berisi informasi tentang banyak account - satu object BankDatabase berpartisipasi dalam relasi komposisi nol atau lebih object Account. Nilai multiplicity 0 .. * di sisi Account dari asosiaso class BankDatabase dan class Account menunjukkan bahwa nol atau lebih object dari class Account ikut serta di dalam asosiasi. Class BankDatabase memiliki sebuah relasi one-to-many dengan class Account - BankDatabase dapat berisi banyak Account. Demikian pula , class Account memiliki relasi many-to-one dengan class BankDatabase – Terdapat kemungkinan banyak Account disimpan dalam BankDatabase.

Ingat dari Gambar 1.8 bahwa nilai multiplicity * identik dengan 0 .. *.

Gambar 1.10 juga menunjukkan bahwa pada waktu tertentu terdapat 0 atau 1 object Withdrawal terjadi. Jika pengguna melakukan Penarikan (withdrawal), “sebuah object dari class Withdrawal mengakses / memodifikasi Saldo rekening melalui sebuah object dari class BankDatabase”. Kita bisa menciptakan asosiasi secara langsung antara kelas Withdrawal dan class Account. Dokumen persyaratan, meskipun, menyatakan bahwa “ATM harus berinteraksi dengan database informasi rekening milik bank” untuk melakukan transaksi. Sebuah rekening bank berisi informasi sensitif, dan pembuat sistem harus selalu mempertimbangkan keamanan data pribadi ketika merancang

sebuah sistem. Dengan demikian, hanya class BankDatabase yang dapat mengakses dan memanipulasi sebuah rekening (account) secara langsung. Semua bagian lain dari sistem harus berinteraksi dengan database untuk mengambil atau memperbarui informasi account (misalnya, saldo rekening).

Class Diagram pada Gambar 1.10 juga memodelkan asosiasi antara class Withdrawal dan class Screen, CashDispenser dan Keypad. Sebuah transaksi penarikan mencakup memberitahu pengguna untuk memilih jumlah penarikan, dan menerima input numerik. Aksi ini memerlukan penggunaan layar dan keypad, secara berurutan. Selain itu, pengeluaran uang kas untuk pengguna membutuhkan akses ke dispenser uang kas.

Class BalanceInquiry dan Deposit, meskipun tidak ditunjukkan pada Gambar 1.10, mengambil bagian dalam beberapa asosiasi dengan class lain dari sistem ATM. Seperti class Withdrawal, masing-masing class berasosiasi dengan class ATM dan BankDatabase. Sebuah object dari class BalanceInquiry juga berasosiasi dengan sebuah object dari class Screen untuk menampilkan Saldo dari account kepada user. Class Deposit berasosiasi dengan class Screen, Keypad, dan DepositSlot. Seperti class Withdrawal, transaksi penyimpanan dana memerlukan penggunaan layar dan keypad untuk menampilkan pemberitahuan dan menerima masukan, secara berurutan. Untuk menerima amplop deposit, sebuah object dari class Deposit mengakses slot penyimpanan. Sekarang kita telah mengidentifikasi class awal dalam sistem ATM kita - dapat menemukan class lain selagi melanjutkan dengan tahap perancangan dan implementasi.

LATIHAN

1. Misalkan kita memiliki class Mobil yang mewakili sebuah mobil. Pikirkan beberapa bagian yang berbeda yang akan disatukan untuk memproduksi mobil secara keseluruhan. Buat sebuah Class Diagram (mirip dengan Gambar 1.9) yang memodelkan beberapa relasi Komposisi dari class Mobil.
2. Misalkan kita memiliki class File yang mewakili sebuah dokumen elektronik dalam computer stand-alone, yang tidak terhubung dengan jaringan, yang diwakili oleh class Komputer. Apa saja jenis asosiasi antara class Computer dan class file?
    a. Class computer memiliki sebuah relasi one-to-one dengan class File
    b. Class computer memiliki sebuah relasi many-to-one dengan class File
    c. Class computer memiliki sebuah relasi one-to-many dengan class File
    d. Class computer memiliki sebuah relasi many-to-many dengan class File
3. Sebutkan apakah pernyataan berikut ini benar atau salah, jika salah, jelaskan alasannya: “Sebuah diagram UML yang di dalamya terdapat ruang ke-2 dan ke-3 tidak dimodelkan disebut sebagai Elided Diagram”.
4. Ubahlah class diagram pada Gambar 1.10 untuk memasukkan class Deposit menggantikan class Withdrawal.