

Modelovanie bankomatového softvéru v UML*

Beáta Belková

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
xbelkovab@stuba.sk

6. október 2021

Abstrakt

Článok sa zaoberá modelom bankomatového softvéru. Na začiatku stručne definuje čo je Unified Modeling Language, ktorý je na modelovanie použitý. Následne pomocou vybraných diagramov demonštruje model softvéru pre bankomat. Použité diagramy sú sekvenčný diagram, diagram tried a diagram prípadov použitia.

Kľúčové slová : bankomatový softvér, model softvéru

1 Úvod

Bankomaty prešli za posledných 60 rokov veľkými zmenami. Z jednoduchých zariadení, ktoré vymieňali jednorazové poukážky za obálky s bankovkami sa stali zariadenia, ktoré požívane denne. Softvér bankomatu sa počas rokov stával čoraz zložitejším, a preto si jeho tvorba vyžaduje sofistikovaný model.

Základné informácie o grafickom jazyku, ktorý je použitý na modelovanie, ako aj popis jeho diagramov sú v časti 2. Príklady vybraných UML diagramov pre bankomatový softvér a ich vysvetlenie článok ponúka v časti 3. Záverečné poznatky a zhrnutie celého článku prináša časť 4.

*Semestrálny projekt v predmete Metódy inžinierskej práce, ak. rok 2021/22, vedenie: Ing. Vladimír Mlynarovič, PhD

2 Unified Modeling Language (UML)

UML je grafický jazyk, ktorý je štandardom v oblasti analýzy a návrhu pri vývoji softvéru. Vo fáze analýzy pomáha odhadnúť cenu systému a taktiež uľahčuje komunikáciu s klientom, keďže diagramy sú jednoduché na pochopenie. Vďaka UML je jednoduchšie reagovať na zmeny v zadaní klienta. Vo fáze návrhu pomáha riešiť otázku ako bude softvér naprogramovaný. [4]

Typy diagramov

UML v súčasnosti pozostáva zo 14 diagramov. Tie sú rozdelené do dvoch základných kategórií: diagramy štruktúry (Structure diagrams) a diagramy správania (Behaviour Diagram).

Diagramy štruktúry

- Diagram tried
- Schéma komponentov
- Objektový diagram
- Zložený štruktúrny diagram
- Schéma balenia
- Schéma nasadenia
- Profilový diagram

Diagramy správania

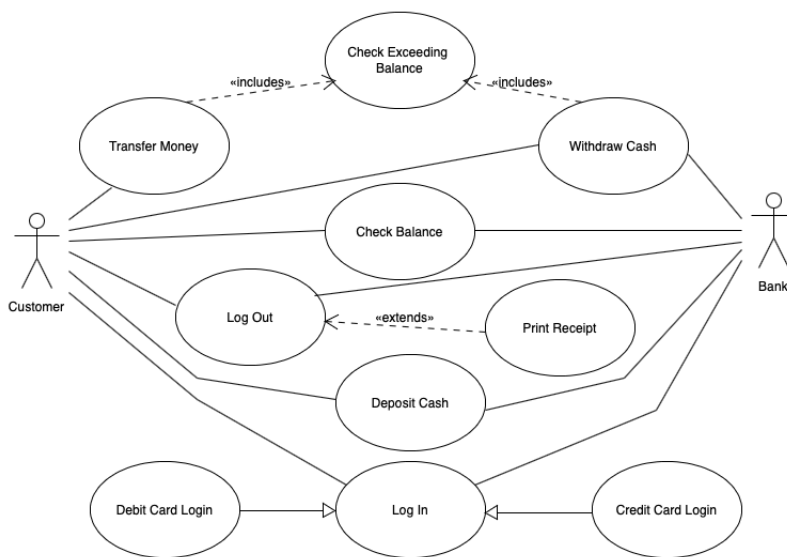
- Diagram aktivít
- Diagram prípadov použitia
- Stavový diagram
- Komunikačný diagram
- Sekvenčný diagram
- Diagram prehľadu interakcií
- Synchronizačný diagram

V článku bude pre model bankomatového softvéru použitý diagram prípadov použitia, diagram tried (obsahuje všetky triedy, ktoré bude softvér používať a vzťahy medzi nimi) a sekvenčný diagram (zobrazuje interakcie medzi zákazníkom, bankomatom a bankou).

3 Model bankomatového softvéru

3.1 Diagram prípadov použitia

Diagram prípadov použitia je väčšinou prvý diagram, ktorý sa pri návrhu systému vytvára. Je to z toho dôvodu, že diagram nerieši ako bude daný systém fungovať, ale čo bude systém vykonávať, teda čo od neho tvorca prípadne klient očakáva. Je zámerne jednoduchý, aby programátor predčasne neuviazol v problémoch s implementáciou systému. Jeho benefitom je, že zobrazuje systém tak, ako ho vidí používateľ a preto uľahčuje komunikáciu s klientom. [6]



Obr. 1: Diagram prípadov použitia [8]

Diagram zobrazuje vzťah medzi aktérmi a systémom a funkcie modelovaného systému. Na obrázku 1 sú prípady použitia znázornené pomocou elipsy, v ktorej sa nachádza popis práce. Čiary spájajúce aktérov a prípady použitia predstavujú interakcie medzi nimi. Komunikácia prebieha zľava doprava (pokiaľ nie je šípkou naznačený iný smer).

Klient môže :

- previezt' peniaze
- vybrať peniaze
- skontrolovať stav na účte
- odhlásiť sa (pod odhlásením sa rozumieme ukončenie transakcie a vytiahnutie karty z bankomatu)
- vložiť peniaze
- prihlásiť sa (pod prihlásením sa rozumieme vloženie karty do bankomatu a zadanie správneho PIN kódu)

Prípady použitia pre prevod a výber peňazí majú na seba väzbou *include* naviazanú funkcionálnu na kontrolu stavu účtu. Táto väzba sa používa ak sa niektorá funkcionálna v rôznych častiach modelu opakuje. Taktiež sa môže použiť keď je nejaká funkcionálna natoľko dôležitá, že bude v diagrame osobitne zobrazená namiesto toho aby bola iba považovaná za súčasť nejakého prípadu použitia. [6]

Ďalším príznačným prvkom diagramu 1 je vzťah *extend* medzi prípadmi použitia pre odhlásenie sa a vytlačenie potvrdenia. Zatiaľ čo väzba *include* naznačuje, že prípad použitia, z ktorého vychádza obsahuje funkcionálnu, na ktorú ukazuje, pri väzbe *extend* to neplatí. Tento vzťah pôsobí opačne. To znamená, že prípad použitia pre odhlásenie sa nepotrebuje k svojej činnosti prípad použitia, ktorý je k nemu naviazaný väzbou *extend*. To však neplatí pre prípad použitia, z ktorého väzba vychádza. Ten nemôže fungovať bez toho aby bol vykonaný prípad použitia, na ktorý ukazuje. [1]

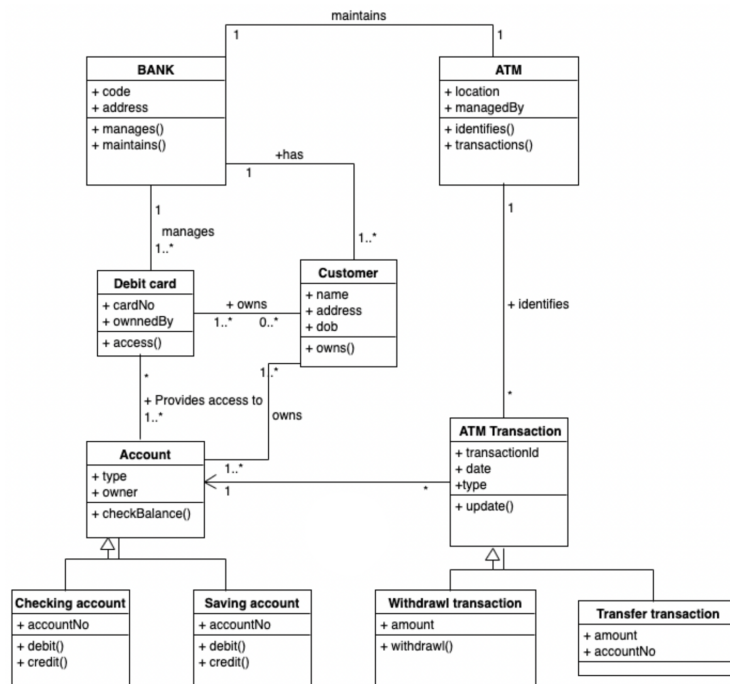
Z diagramu 1 sa dá teda určiť, že odhlásenie sa vykoná bez ohľadu na to či sa vytlačí potvrdenie. Avšak potvrdenie sa vytlačí iba v tom prípade, že sa používateľ odhlási.

Posledný špecifický vzťah v diagrame je vzťah *zovšeobecnenia*, ktorý reprezentuje hrana so šípkou. Označuje, že prípad použitia dedí funkcionálnu nadradeného prípadu použitia. Šípka ukazuje na nadradený prvok. Diagram teda demonštruje prípad, kedy sa pri odhlasovaní vyberie prípad použitia buď pre kreditnú alebo debetnú kartu a tie zdedia všetky metódy a funkcie pre odhlásenie. [6]

Druhým aktérom v diagrame 1 je banka. Je to z toho dôvodu, že pri vykonávaní akéhokoľvek prípadu použitia musí bankomat komunikovať s bankovou inštitúciou, ktorá mu poskytuje všetky potrebné dáta o klientovi. Po uskutočnení operácie taktiež dáta aktualizuje a odovzdá banke.

3.2 Diagram tried

Diagram tried je najpoužívanejší UML diagram. Zobrazuje triedy, ktoré bude softvér obsahovať a vzťahy medzi nimi. Diagram musí byť kompletný aby sa podľa neho dal napísať kód a preto musia triedy obsahovať všetky atribúty a metódy. Pod pojmom atribúty si môžeme predstaviť dáta, s ktorými bude trieda pracovať. Metódy sú operácie, ktoré bude daná trieda vykonávať. [7]



Obr. 2: Diagram tried [8]

Na obrázku 2 je ako prvá zobrazená trieda **Bank**. Atribúty, ktoré bude obsahovať sú kód na jej identifikáciu a adresa. Metódy, ktoré banka používa na spoluprácu s bankomatom sú spravovanie a údržba. Znamienko plus pred názvami atribútov a metód znamená, že sú verejné.

Ďalšou triedou v diagrame je **ATM**, čiže bankomat. Metódy tejto triedy deklarujú transakcie, ktoré vykonáva.

Ako uvádza článok [2], každá banka potrebuje klientov, a preto diagram obsahuje triedu **Customer**. Údaje, ktoré trieda potrebuje sú meno klienta a adresa. Klient nevykonáva žiadne operácie iba vlastní produkty banky.

Klient môže vlastniť účet a debetnú kartu. Preto bude diagram obsahovať triedy **Account** a **Debit card**. Účet má v atribútoch zapísaného vlastníka a typ účtu. Táto trieda môže vykonať kontrolu stavu na účte. Trieda pre debetnú kartu má v atribútoch zapísané číslo karty a vlastníka karty. Metóda karty je "prístup", pretože umožňuje prístup k bankomatu a tým aj k účtu vlastníka.

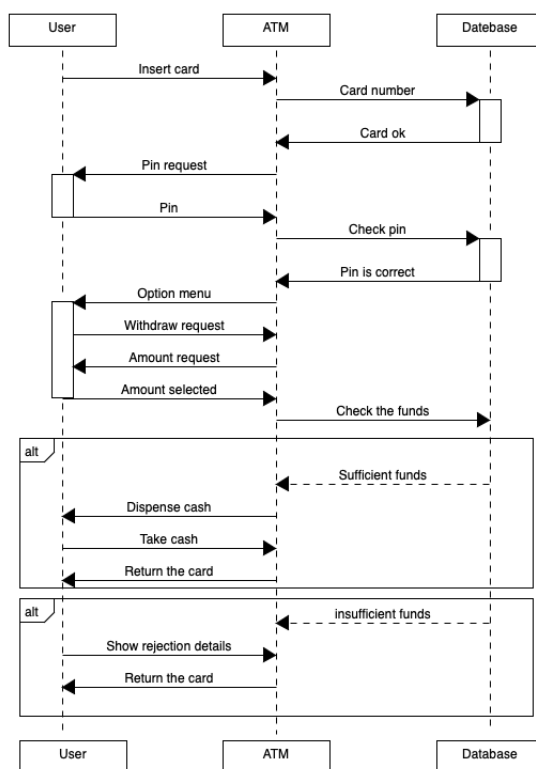
Ďalšie triedy, ktoré musí diagram obsahovať popisujú transakcie vykonávané počas používania bankomatu. Trieda **ATMtransaction** má tri atribúty – ID transakcie, dátum a typ transakcie. Jej jedinou metódou je aktualizácia.

V prípade diagramu 2 sa predpokladá, že zákazník môže iba vybrať peniaze alebo peniaze previesť na iný účet. Preto obsahuje triedy **WithdrawalTransaction** a **TransferTransaction**. Tieto dve triedy sú podtriedami **ATMtransaction** a

preto môžu zdediť jej metódy. Transakcia výberu potrebuje údaje o sume, ktorú chce klient vybrať. Potom metóda výber uskutoční vydanie hotovosti klientovi a zdedená metóda aktualizácie aktualizuje stav na účte. Transakcia prevedenia peňazí na druhý účet má iba dva atribúty. Sumu, ktorú chce klient poslať a číslo účtu, na ktorý má byť suma pripísaná. Trieda po schválení transakcie využije zdedenú funkciu aktualizácia, ktorá aktualizuje stav na účte. [2]

3.3 Sekvenčný diagram

Posledný diagram, ktorému sa bude článok venovať je sekvenčný diagram. Na základe diagramu tried už vieme aké triedy bude softvér obsahovať a aj aké vzťahy budú medzi nimi. Sekvenčný diagram teraz popíše chronologickú komunikáciu medzi týmito triedami a objektami v čase. Diagram teda zobrazuje ako, a v akom poradí skupina objektov spolupracuje. V sekvenčnom diagrame nie sú kľúčové len kroky aktivity, ale aj vzájomné vzťahy medzi triedami a objektami, ktoré v ňom figurujú. [5]



Obr. 3: Sekvenčný diagram [3]

Diagram 3 simuluje výber hotovosti z bankomatu. V hornej časti diagramu sa nachádzajú symboly objektov (obdĺžniky), ktoré reprezentujú aktérov. V tomto prípade budú v diagrame figurovať Klient, Bankomat a Databáza (súčasť bankovej inštitúcie, ktorá poskytuje bankomatu údaje o klientovi). Vertikálne prerušované čiary predstavujú plynutie času. Začínajú tvarom obdĺžnika alebo postavou herca. [3]

Klient najskôr vloží platobnú kartu do bankomatu. Kor znázorňuje prvú interakcia na diagrame. Interakcie je znázornená plnou šípkou s plnou čiarou. Tento symbol sa používa keď odosielateľ musí čakať na odpoveď. Následne musí bankomat overiť, či je číslo karty platné, a preto pošle kvôli overeniu správu do databázy. Pri databáze sa vytvorí aktivačná lišta, ktorá ukáže, že databáza je počas tohto obdobia aktívna. Po overení pošle bankomatu správu, že je karta v poriadku. Táto správa bude odoslaná späť na konci aktivačnej lišty [3]

Teraz bankomat požiada používateľa aby zadal svoj PIN kód. Po zadaní kódu musí bankomat opäť skontrolovať databázu a overiť, či je kód správny. Tá následne PIN kód overí a pošle bankomatu správu. Pri overovaní údajov v databáze bude na diagrame opäť znázornená aktivačná lišta. Keď je všetko v poriadku bankomat ponúkne klientovi ponuku možností.

V tomto bode prebieha výmena interakcií medzi klientom a bankomatom. Klient si z ponúknutých možností zvolí možnosť výberu hotovosti, bankomat sa spýta na čiastku a klient zadá čiastku, ktorú chce vybrať. Počas celej interakcie je pri klientovi znázornená aktivačná lišta.

Ako pri predošlých interakciách aj teraz musí bankomat poslať správu do databázy a overiť, či má klient na účte dostatok prostriedkov. Teraz však diagram zobrazí oba prípady odpovede. Aby bolo z diagramu zreteľne jasné, aké interakcie budú v oboch prípadoch prebiehať, sú znázornené oddelene. Pre oddelenie viacerých možností odpovede sa v sekvenčnom diagrame používajú slučky (prázdne obdĺžniky, ktoré majú v pravom hornom rohu štítok pre text). V prípade diagramu 3 je v štítoku text alt = alternatíva. [3]

Ak má klient na účte dostatok prostriedkov bankomat vydá hotovosť, klient si ju vezme a bankomat mu nakoniec vráti platobnú kartu (v niektorých prípadoch bankomatom vracia kartu pred vydaním hotovosti, takže interakcie v tejto časti budú na diagrame vymenené).

V opačnom prípade databáza pošle bankomatu správu o tom, že klient nemá na účte dostatok prostriedkov. Bankomat následne zobrazí klientovi správu o zamietnutí transakcie a vráti mu platobnú kartu.

4 Záver

Literatúra

- [1] RNDr. Ilja Kraval. Kdy použít v use case diagramu vztah extend a kdy include (část 1). https://www.objects.cz/wp/is_uml/kdy-pouzit-v-use-case-diagramu-vztah-extend-a-kdy-include/.
- [2] Constantine Nalimov. Class diagram for an atm system: step-by-step guide. <https://www.gleek.io/blog/atm-system-class.html>.
- [3] Constantine Nalimov. Sequence diagram for atm withdrawal: a step-by-step guide. <https://www.gleek.io/blog/atm-sequence-diagram.html>.
- [4] David Čápka. 1. diel - Úvod do uml. <https://www.itnetwork.sk/navrh/uml/uml-uvod-historie-vyznam-a-diagramy>.
- [5] David Čápka. 10. diel - uml - sequence diagram. <https://www.itnetwork.sk/navrh/uml/uml-sequence-diagram>.
- [6] David Čápka. 2. diel - uml - use case diagram. <https://www.itnetwork.sk/navrh/uml/uml-use-case-diagram>.
- [7] David Čápka. 5. diel - uml - class diagram. <https://www.itnetwork.sk/navrh/uml/uml-class-diagram-triedny-model>.
- [8] DAV University. Sample of uml diagrams for atm system. <https://www.davuniversity.org/images/files/study-material/UML-ATM.pdf>.