VŠB - Technická univerzita Ostrava

Objektově orientované technologie

Daniela Szturcová

Úvod

Výuková podpora pro cvičení z předmětu Objektově orientované technologie vznikla za podpory projektu Inovace bakalářských a magisterských studijních oborů na Hornicko-geologické fakultě VŠB-TUO pod číslem CZ.1.07/2.2.00/28.0308. Tento projekt je realizován za spoluúčasti EU.

Cíle jednosemestrální výuky lze shrnout do několika bodů:

- Vyzkoušet principy objektových technologií při tvorbě vlastního modelu systému.
- Vytvářet základní diagramy modelovacího jazyka UML.
- Porozumět a interpretovat základní diagramy UML.
- Naučit se postup při tvorbě modelu systému.

V předkládaném materiálu se studenti naučí, jak postupovat při tvorbě modelu systému. Budování modelu je postaveno na principech objektově orientovaných technologií a využívá standardu UML 2. V teoretické části cvičení jsou zopakovány základní pojmy, jejichž připomenutí pomáhá při tvorbě diagramů a dokumentace spojené s vytvářením modelu systému. Předpokládá se obeznámení s tématy již z přednášek, zde se nalézá pouze stručné shrnutí. Dále následuje popis postupu, který je na konkrétním zadání systému použit. Postup si student zopakuje při samostatné práci na zadání jiného systému.

Počítačovou podporou, kterou při tvorbě budeme využívat, je momentálně jeden z CASE nástrojů – aplikace Visual Paradigm for UML, jejíž školní licence je studentům po dobu školního roku zdarma. Na stránkách produktu lze najít celou řadu pomůcek a tutoriálů (http://www.visual-paradigm.com/tutorials/?category=umlmodeling). Licence produktu je pravidelně obnovována, lze si stáhnout odpovídající verzi produktu z webových stránek VP a klíč pro spuštění verze StandardEdition studenti obdrží na cvičení.

1 Specifikace systému

Cíl cvičení

Vypracovat specifikaci systému.

1.1 Teoretický základ

Specifikací systému rozumíme popis struktury a chování systému. Specifikaci lze vytvořit formální nebo neformální. Pro popis systému je nutno znát doménu, kterou se systém zabývá. V případě známé domény je možné nadefinovat účel systému a jeho popis vytvořit formou textového zápisu přímo. V případě neznalosti je nutno nejprve provést sběr informací, ověřit jejich správnost a pochopení.

Při sběru informací o systému je možné použít různé metody zjišťování údajů:

- interview účastníky,
- dotazníková šetření,
- studium dokumentů, keré již existují a lze je po dohodě se zadavatelem získat,
- pozorovat reálné procesy nebo se přímo pracovních procesů zúčastnit a provést záznam.

Analýzou specifikace vznikají požadavky na systém, které zaznamenáme do podoby, která je srozumitelná jak realizačnímu týmu, tak i zadavateli a budoucím uživatelům systému.

1.2 Obsah cvičení

Porozumět již existující specifikaci a na základě porozumění vytvořit podkladový dokument pro další fáze tvorby modelu.
Vytvořit specifikaci na základě znalostí zvolené domény.

1.3 Úkoly

Vytváření dokumentace

Níže uvedené ukázky specifikací systému zpracujeme do podoby dokumentu, ve kterém budou uvedeny rozeznané prvky systému, jejich vztahy a předpokládané chování jednotlivých prvků.

- Systém pro nákup plechovkových nápojů *Nápojový automat*.
- Systém pro jízdu taxíkem *Taxis*.

Pro každou specifikaci:

1. Prostudujte specifikaci a pokuste se rozeznat prvky systému.

2. Vytvořte seznam prvků systému, u každého prvku nadefinujte relevantní vlastnosti:
   - výstižně atribut pojmenujte,
   - uveďte doménu atributu,
   - omezení, která ze specifikace systému pro atribut plynou,
   - případně určete a zaznamenejte datový typ.
3. Uveďte, které prvky jsou propojeny vztahem, vztah popište.
4. Pro každý rozeznaný prvek specifikujte chování, které v rámci systému vykazuje.

Pro splnění úkolů se dotazujte pedagoga, který bude upřesňovat obsah specifikace. Debatujte se spolužáky a prohlubujte znalost domény a porozumění systému.

Dokument vytvořte formou tabulky, kde v řádcích budou uvedeny názvy prvků a ve sloupcích požadované parametry.

Ukázka specifikace – Nápojový automat

Účel

Nápojový automat je zařízení, které zákazníkům nabízí koupi nealkoholického nápoje v plechovce.

Specifikace

Nápojový automat je naplněn různými druhy nápojů. Jejich nabídka je viditelná pro zákazníka ve formě označení nápoje, druhu nápoje a ceny. Zákazník přistupuje k nápojovému automatu, vybere si nápoj z nabídky. Aby mu mohl být nápoj vydán, je vyzván k vhazování mincí do otvoru pro mince. Po vyhodnocení dostatečného obnosu nápojový automat vhodí do výdejního boxu pro nápoj zaplacený nápoj. V případě přeplatku se do boxu na mince vrátí přeplatek.

Nápojový automat je v provozu udržován pracovníkem provozovatele, který doplňuje nebo vyměňuje nápoje, vybírá pokladnu s mincemi a doplňuje příslušné mince na vracení přeplatku.

Nápojový automat je v provozu 24 hodin, 7 dní v týdnu. Je připojen do elektrické sítě a je uzavřen do ochranného boxu, který zamezuje krádeži.

Ukázka specifikace – TaxiS

Účel

TaxiS je systém zaměřený na optimalizaci jízd taxíků v rámci firmy TaxiS. Cílem systému je umožnit jak řidičům, tak zákazníkům snížit náklady na jízdy taxíkem.

Specifikace

TaxiS bude mít návaznost na systém objednávek jízd, dispečink ap. V systému lze rozeznat několik různých typů objektů, které se účastní činností systému. Stručně je uvedena jejich charakteristika.

Vozidla Systém eviduje vozidla, která se mohou nacházet ve dvou stavech: aktivní, neaktivní. Aktivní – v daném okamžiku jsou k dispozici zákazníkům, neaktivní – nejsou provozuschopná – mohou být v servise. U vozidel se eviduje SPZ, typ, označení, rok výroby, datum platnosti STK, počet pasažérů, objem zavazadlového prostoru, datum předpokládaného servisu.

Řidiči Firma má seznam řidičů, kteří jsou buď zaměstnanci firmy, nebo je možné s nimi sjednávat externí spolupráci. Sledují se jejich jméno, příjmení, datum narození, adresa, platnost a typ oprávnění ŘP.

Zákazníci Naše služby využívá několik stálých zákazníků, kteří díky objemu již realizovaných obchodů mají speciální ceny. Dále jsou to jednorázoví zákazníci. Evidujeme jméno a příjmení, v případě firmy její název, kontaktní telefon.

Dispečink Jedná se o operátory, kteří pomáhají sledovat rezervace a spojují se s řidiči na trase, aby jízdy byly pokud možno co nejefektivněji realizovány. Evidujeme jejich jméno, příjmení, datum narození a adresu v systému.

Rezervace jízdy Existuje několik kanálů. které umožňují rezervovat či objednat jízdu taxíkem:

- přes Internet pomocí objednávkového formuláře,
- telefonicky,
- e-mailem na adrese taxi@taxi.cz,
- pomocí sms.

Každé objednávce jízdy je přiřazeno jednoznačné číslo. U objednávky musí být uvedeno jméno zákazníka, kontakt na zákazníka – telefon, místo odvozu, cílové místo, datum a hodina, kdy má být vůz přistaven. Stejně tak je možné jízdu realizovat pro zákazníka, který si taxi "chytne" na ulici bez předchozí rezervace.

Jízda Jízda vozidlem naší firmy je evidována pomocí těchto údajů: datum jízdy, čas odjezdu z místa odvozu a čas příjezdu do cíle, místo, kde byl zákazník naložen, cílové místo, doba čekání během jízdy, počet najetých kilometrů dle tachometru. U každé jízdy se vypočítává cena za jízdu – jedná se o součet následujících položek:

Cena = (nástupní sazba + počet najetých km + sazba dle typu zákazníka + počet minut čekání) × sazba za čekání.

Sazba Eviduje se několik typů různých sazeb:

- základní,
- nadstandard,
- obousměrná,

- do zahraničí,
- VIP,
- z ulice.

Samostatný úkol

Vytvořte specifikaci domény, kterou dobře znáte. Zopakujte výše uvedené body a vytvořený dokument odevzdejte.

2 Požadavky na systém

Cíl cvičení

Vyhledat požadavky na systém a vytvořit jejich specifikaci.

2.1 Teoretický základ

Požadavek – vlastnost, kterou vyžadujeme od systému. Pro uživatele/účastníka systému představuje hodnotu, kterou od systému očekává.

Specifikace požadavků na systém se sestává ze dvou skupin požadavků:

- *funkční požadavky* – určují chování, které má navrhovaný systém nabízet uživatelům,
- *nefunkční požadavky* – lze chápat jako omezení systému z různých hledisek (legislativní, technické apod.).

Funkční požadavky budou zachyceny v modelu případů užití, nefunkční je nutno modelovat v modelu požadavků, který ovšem v UML nemá podporu. [1]

2.2 Doporučený postup při vyhledávání požadavků

Vyhledávání požadavků na systém je možné provézt v následujících krocích.

1. Z textové specifikace systému vybereme všechny popisy činností, které od systému očekávají různí uživatelé.
2. Na základě významu vyhledaných činností rozlišíme, které z nich představují požadovanou funkci systému.
3. Vybranou činnost výstižně pojmenujeme a zařadíme do seznamu požadavků.

2.3 Úkoly

1. Vytvořte seznam uživatelů systému - kdo bude systém užívat?
2. Ve specifikaci systému vyhledejte chování/funkce, které se od systému očekávají pro jednotlivé uživatele. Názvy funkcí volte co nejvýstižněji tak, aby bylo patrné očekávání uživatele.
3. Setřiďte požadavky do skupin podle uživatelů.
4. Vyhledejte požadavky, které spolu souvisí – vykazují buď shodu, nebo představují opačné „strany jedné mince“ a vyznačte souvislost mezi nimi. (například vhoditNapojDoVydejniku/odebratNapoj).

Seznam požadavků – Nápojový automat

Funkční požadavky

Zakaznik – během nákupu nápoje bude požadovat:

- možnost výbrat si nápoj,
- vkládat mince (přijímat mince),
- odebírat přeplatek,
- odebírat nápoj.

Automat – bude reagovat pomocí svých funkcí:

- přijímat mince (vkládat mince),
- vydávat nápoj,
- vracet přeplatek.

Zasobovac – manipuluje s nápoji a očekává:

- možnost doplnit nápoje do zásobníku nápojů,
- kontrolovat expiraci nápoje,
- odebrat nápoj ze zásobníku nápojů.

Pokladnik – manipuluje s penězi a požaduje funkce:

- doplnit mince do mincovníku,
- odebrat mince z pokladny.

Rozhraní – komunikace se zákazníkem:

- hlásí stav zásobníku a reaguje na požadavky zákazníka.

Nefunkční požadavky

- Zajistit 24 hodinový provoz.
- Po odběru nápoje možnost opětovného nákupu po desetisekundové pauze.
- Zabezpečit zařízení proti krádežím nápojů a mincí – provedení s možností přístupu autorizovaným osobám, které manipulují s nápoji a s mincemi.

Samostatný úkol

Vypracujte seznam požadavků pro systém Taxis.

3 Tvorba diagramu případů užití

Cíl cvičení

Vyhledat aktéry, hranice systému a pro každého aktéra jeho případy užití. Vytvořit diagram případů užití.

3.1 Teoretický základ

Model případů užití se vytváří v počátečním stádiu analýzy. Pomáhá zachytit funkční požadavky na systém. Tvoří jej diagramy případů užití, specifikace případů užití, specifikace aktérů.

Diagram případů užití (Use Case Diagram) popisuje chování systému z pohledu uživatelů systému. Znázorňuje, jaké typy uživatelů a při jakých příležitostech používají systém. Funkční požadavky na systém jsou zde zachyceny formou interakcí uživatele a systému, tj. jaké je chování/reakce systému v případě požadavku konkrétního uživatele. Každé užití systému uživatelem je chápáno jako činnost, která je definována svým názvem a jejíž obsah je popsán soustavou postupně vykonávaných kroků (scénářem).

Dle [1] spočívá modelování případů užití v následujících krocích:

- nalezení hranic systému,
- vyhledání aktérů,
- nalezení případů užití,
- specifikace případů užití,
- určení alternativních scénářů,
- opakování předchozích bodů, dokud nedojde k ustálení případů užití, aktérů a hranic systému.

V diagramu případů užití modelujeme prvky diagramu pomocí následujících symbolů:

Aktér představuje uživatele systému. Jedná se o osobu nebo systém, případně i čas (nastane specifická událost), kdy komunikují s naším systémem, stojí mimo náš systém. Název aktéra vystihuje, v jaké roli vůči systému vystupuje. Budeme rozlišovat aktéry hlavní a vedlejší. Znázorňují se symbolem panáčka a názvem.

Hranice systému stanoví rozsah systému a pomáhají uvědomit si různá rozhraní dle interakcí aktérů a systému. Znázorňuje se rámečkem kolem množiny případů užití s uvedením názvu modelovaného systému.

Případ užití představuje určitou funkcionalitu systému. Aktér ji využívá pro splnění svého cíle. Postup použití (sekvence interakcí) je podrobněji uveden ve scénáři. Název volíme z pohledu aktéra a co nejvýstižněji dle požadavané funkcionality. Pro případ užití se užívá symbol elipsy, uvnitř je uveden název případu užití.

Asociace znázorňuje komunikaci mezi aktérem a případem užití.

Zahrnutí (include) představuje vztah mezi dvěmi případy užití, kdy chování popsané v jednom případem užití je součástí v jiném případu užití. Základní případ užití je závislý na vkládaném, bez něj nelze realizovat celou interakci. Reprezentuje jej přerušovaná čára, která spojuje případy užití se šipkou směrem k závislému případu užití (zde je hlavní závislý na vloženém) s uvedením stereotupu ≪include≫.

Rozšíření (extend) představuje vztah mezi dvěmi případy užití, kdy chování popsané v jednom případem užití je rozšířením interakcí v jiném případu užití. Základní případ užití může a nemusí obsáhnout vkládananý, rozšiřující případ užití. Reprezentuje jej přerušovaná čára, která spojuje oba případy užití se šipkou směrem k závislému případu užití (zde je rozšiřující závislý na hlavním) s uvedením stereotupu ≪extend≫.

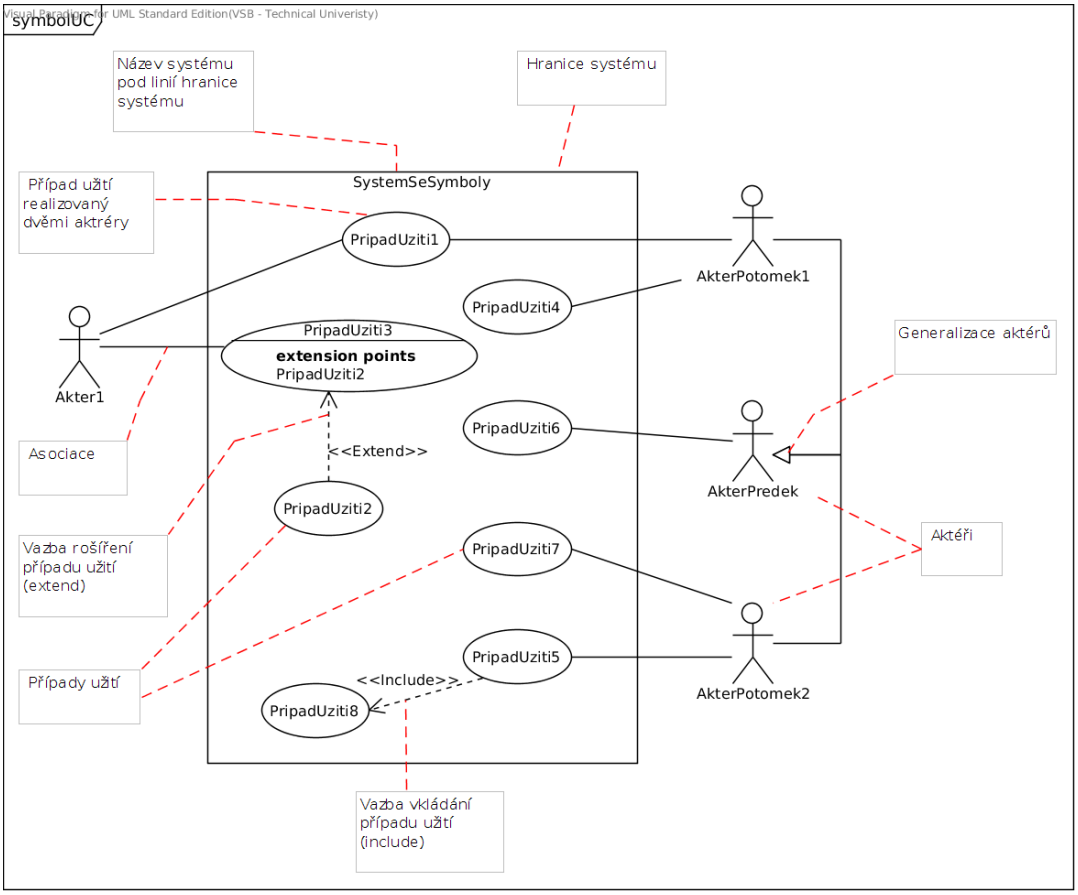
Generalizace se používá v případě, že několik aktérů nebo případů užití má podobné chování, které se dá zobecnit. Generalizace se modeluje šipkou s trojúhelníčkem na konci u obecnějšího prvku.

Prvky jazyka UML a jejich vzájemné vztahy v diagramu případů užití jsou zakresleny a popsány na obrázku 1.

3.2 Doporučený postup

Při tvorbě modelu případů užití budeme postupovat v těchto krocích:

1. Nalezneme a vymezíme hranici systému.
2. Specifikujeme aktéry (případně provedeme kategorizaci na primární, vedlejší).
3. Specifikujeme případy užití systému jednotlivými aktéry.
4. Vytvoříme diagram případů užití.
5. Ke každému případu užití vytvoříme hlavní scénář, který zachytí interakce aktéra a systému. (Práce se scénáři je podrobněji rozebrána v následujícím cvičení.)
6. Popíšeme alernativy k hlavnímu scénáři (alternativa úspěšná, alternativa chybová včetně ošetření chyb).
7. Nalezneme případy užití, které je možno vyčlenit – vazby include.
8. Navrhneme případy užití rozšiřující hlavní scénář, uvedeme body rozšíření.
9. Diagram případů užití upravíme o případné další případy užití.



Obrázek 1: Obrázek symbolů diagramu případu užití (vytvořeno v akademické licenci Visual Paradigm).

3.3 Úkoly

Vytváření diagramu

Pro vytvoření modelu případů užití pro systém *Nápojového automatu* je nutno sestavit seznam aktérů, případů užití a jejich vztahů. Dále uvedené kroky napomáhají postupu, který vede k vytvoření počáteční dokumentace a diagramu případu užití v prostředí podpůrného nástroje pro tvorbu diagramů.

Vyjdeme ze specifikace systému a ze seznamu požadavků na systém, doplníme na základě naší zkušenosti s použitím nějakého nápojového automatu.

1. Nalezení hranic systému u nápojového automatu lze stanovit pomocí představy automatu jako plechového boxu, který obsahuje nápoje, jejich zásobník, pokladnu, mincovník, atd.

2. K vyhledání a upřesnění aktérů použijeme návodné otázky.

Kdo (nebo co) používá systém? Systém používají

- zájemci o nápoje, Zakaznik
- pracovník obsluhy, který doplňuje nápoje, Zasobovac
- pracovník obsluhy, který výbírá pokladnu a doplňuje mince, Pokladnik

Jakou roli hraje při této interakci?

- Zakaznik – přistupuje k automatu s úmyslem koupit si nápoj. (zvolit si nápoj, vkládá mince, odebírá vybraný nápoj, odebírá mince)
- Zasobovac – stará se o dostatek nápojů v automatu a jejich čerstvost. (vkládá nové nápoje, vybírá staré nápoje, kontroluje funkčnost automatu)
- Pokladnik – vybírá shromážděné mince, doplňuje drobné na vracení přeplatků.

Kdo systém instaluje?

Instalaci provádí pracovník firmy na nápoje – Prodejce.

Kdo systém spouští?

Systém uvádí do provozu pracovník firmy na nápoje – Prodejce.

Kdo systém vypíná?

Systém vypíná pracovník firmy na nápoje – Prodejce.

Kdo se stará o údržbu systému?

Systém udržuje v chodu Zasobovac a Pokladnik.

Existují jiné systémy, které spolupracují se systémem? Jaké systémy to jsou?

Kromě přívodu elektrické energie automat nepotřebuje spolupracovat s dalšími systémy. (Dalo by se uvažovat o poslání zprávy, například formou sms či signálu na centrální evidenci, v případě přeplnění pokladny, odebrání všech nápojů nebo fatální chyby automatu.)

Kdo do systému zadává informace? Kdo tyto informace používá?

- Zakaznik – vložením mincí se přičte jejich hodnota do pokladničky.
- Zasobovac – podle počtu kusů jednotlivých nápojů, které do automatu doplnil, se navýší jejich počet u sčítače nápojů.
- Pokladnik – zadá počet kusů jednotlivých mincí při doplnění mincí do zásobníku mincí.

Kdo odebírá informace ze systému?

- Zakaznik – zjišťuje, zda je k dispozici zvolený nápoj, případně si čte výzvu ohledně vložení mince a její hodnoty, odebráním nápoje se změní počet kusů daného nápoje, odebráním přeplatku se mění počet mincí v zásobníku mincí.
- Zasobovac – počet kusů jednotlivých nápojů, které z automatu vyjmul.

- Pokladnik – počet kusů a hodnoty jednotlivých mincí z pokladny, v případě doplnění zásobníku mincí zjišťuje jejich počet před doplněním.

Existují nějaké události v systému (nebo se systémem) v určitou dobu?

Časově závislé funkce systém nemá. (Dalo by se uvažovat o pravidelném doplňování nápojů a výběru mincí v případě vyhodnocení odběru nápojů po určité době provozu.)

3. Pro specifikaci případů užití systému jednotlivými aktéry nám mohou pomoci kontolní otázky. Využijeme již hotových úvah z minulého cvičení a rozboru akcí aktérů z minulého úkolu.

Jaké funkce od systému očekává každý účastník? CO systém aktérům umožní?

- Zakaznik – objednat nápoj, vhodit mince, odebrat si požadovaný nápoj, odebrat mince (v případě plné pokladničky, nebo vracení drobných v případě menší ceny za nápoj než je celková hodnota vhozených mincí),
- Zasobovac – doplňovat nápoje, odebírat nápoje. kontrolovat expiraci napojů,
- Pokladnik – doplňovat a odebírat mince.

Bude systém uchovávat nebo poskytovat nějaké informace? Kteří účastníci budou aktivovat činnost?

- Typ nápoje – bude zveřejňován *Zakaznikovi*,
- počet nápojů – pro *Zasobovace*, pro vnitřní dopočet po odebrání, případně dodání nápojů,
- typ a počet mincí uložených v zásobníku automatu – ty budou pouze pro *Pokladnika*,
- pokladnička mincí – kolik a jaká hodnota mince byla vhozena zákazníkem, určeno pro *Pokladnika*.

Který z účastníků bude upozorněn na případnou změnu stavu systému?

Zakaznik - v případě nedostatku nápojů, přeplnění zásobníku na mince, jiné chyby znemožňující funkčnost.

Pokladnik, Zasobovac - při napojení systému do „nějaké“ komunikační sítě by mohl automat odeslat informaci o vyprázdnění zásobníku nápojů a mincí či přeplnění pokladničky na mince, případně hlášení nějaké poruchy.

Existují nějaké vnější události, které ovlivňují systém? Čím (jak) je systém na tyto události upozorněn?

Vypnutí přívodu elektrické energie, vykradení. Kontrolka napětí elektrického proudu, zájemce o automat. Nebudeme dále řešit.

Výstupem tohoto bodu by měl být seznam případů užití a přiřazení k jednotlivým aktérům.

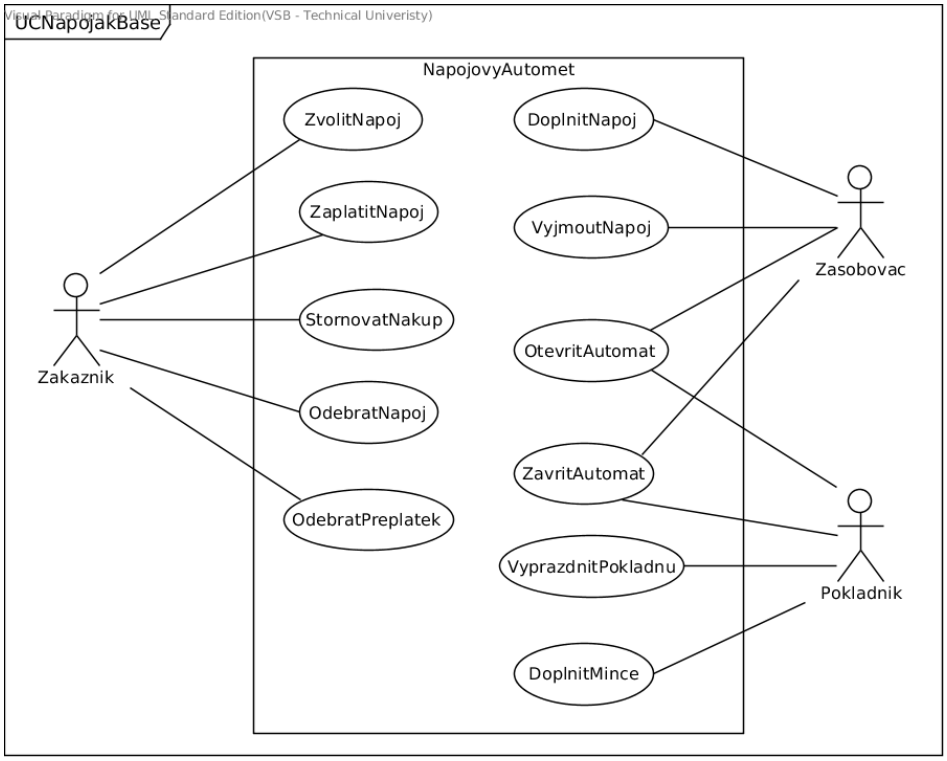
Seznam případů užití (Nápojový automat)

UC	Název UC	Aktér
UC1	ZvolitNapoj	Zakaznik
UC2	ZaplatitNapoj	Zakaznik
UC3	OdebratNapoj	Zakaznik
UC4	OdebratPreplatek	Zakaznik
UC5	StornovatNakup	Zakaznik
UC6	OtevritAutomat	Zasobovac, Pokladnik
UC7	ZavritAutomat	Zasobovac, Pokladnik
UC8	VyprazdnitPokladnu	Pokladnik
UC9	DoplnitMince	Pokladnik
UC10	DoplnitNapoje	Zasobovac
UC11	VyjmoutNapoje	Zasobovac

Obrázek 2: Ukázka seznamu případů užití a vztažených aktérů.

4. Po nalezení hranic, aktérů a případů užití můžeme přistoupit k tvorbě diagramu případů užití.
   a) Spustíme grafický editor pro tvorbu UML diagramů. V našem případě se jedná o *VisualParadigm for UML*, další kroky nastiňují postupné vytváření prvního diagramu.
   b) Po dotazu na adresář pro dočasné soubory se otevře okno aplikace.
   c) Vytvoříme nový projekt s názvem Napojak.
   d) Z levého seznamu diagramů vybereme UC diagram, pravým tlačítkem klikneme a zvolíme Vytvořit nový UC diagram.
   e) Zvolíme symbol pro System - tím znázorníme hranice systemu. Doplníme název systému.
   f) Podle připraveného seznamu aktérů vytvoříme jednotlivé aktéry a symboly s patřičným názvem umístíme mimo hranici systému.
   g) Zvolíme symbol *Případu užití* a pojmenujeme jej dle názvu z našeho seznamu případů užití. Pokud máme aktivní symbol Případu užití, zvolíme ikonku aktéra a přetáhneme jej k aktérovi, kterému je tento případ užití přiřazen v našem seznamu. Tím je vytvořena asociace mezi aktérem a případem užití.
   h) Asociaci lze také vytvořit následovně: při aktivním symbolu aktéra zvolíme ikonu Případu užití a a přetáhneme jej k Případu užití, který aktér požaduje.
   i) Opakujeme tak dlouho, až vyčerpáme všechny aktéry, případy užití a asociace z našich připravených seznamů z předešlých úkolů.

První verze diagramu případu užití je předvedena na obrázku 3.



Obrázek 3: Ukázka seznamu případů užití a vztažených aktérů.

Samostatná práce

Vypracujte diagram případů užití pro systém Taxis a převeďte jej cvičícímu.

4 Tvorba scénářů

Cíl cvičení

Vytvořit scénáře pro případy užití, zachycující průběh interakcí aktérů a systému v případech užití.

4.1 Teoretický základ

Scénářem rozumíme textový popis v podobě sekvencí kroků, které detailněji popisují případ užití. V UML není uveden standardizovaný způsob, jak má scénář vypadat a co všechno by měl obsahovat. Lze ovšem použít doporučovaných šablon od různých autorů, například [1] nebo si můžeme vytvořit vlastní šablonu. Doporučuje se členit šablonu do určitých částí, které zaručují podstatný informační obsah a dodržet časový sled popisovaných kroků.

Doporučují se následující části, které by měl každý scénář obsahovat:

- *ID případu užití* – identifikátor případu užití.
- *Název případu užití* – přenese se název použitý v diagramu případů užití (jiné pojmenování vede k nekonzistenci).
- *Stručný popis případu užití* – stručná specifikace podstaty případu užití.
- *Hlavní (primární) aktér* – aktér, který spouští případ užití, tj. v prvním kroku scénáře zahajuje interakci se systémem.
- *Vedlejší (sekundární) aktéři* – aktéři, kteří jsou v průběhu scénáře zainteresováni do případu užití.
- *Vstupní podmínky* – podmínky, které musí být splněny před zahájením případu užití.
- *Následné podmínky* – podmínky, které musí být splněny, pokud případ užití úspěšně proběhl.
- *Scénář* – sekvence kroků/interakcí případu užití. Hlavní scénář představuje ideální variantu průchodu případem užití. Je možné použít větvení zaznamenané pseudokódem, případně odkazy na jiné případy užití.
- *Alternativní toky* – kroky, které se provádějí při výjimkách či podmíněných situacích (průchod mimo hlavní scénář za určitých podmínek).

Hlavní scénář by měl být stručný, doporučuje se maximálně 10 kroků. Při psaní scénáře je nutno popisovat kroky z pohledu aktéra. Je vhodné použít stručné a jednoduché věty v deklarativní formě s využitím pseudojazyka. Zde je možné využít jednoduchých příkazů pro podmínku (JE-LI, PAK, JINAK) a pro cyklus (DOKUD/POKUD).

UC: ZaplatitNapoj

ID: UC2

Stručný popis:
Zákazník zaplatí za vybraný nápoj.

Hlavní aktéři:
Zákazník

Vedlejší aktéři:
Nejsou

Vstupní podmínky:
Automat přijal požadavek Zakaznika.

Následné podmínky:
Automat je připraven vydat nápoj.

Hlavní scénář:

1. Zákazník si přečte cenu za zvolený nápoj.
2. Automat vyzve zakaznika, aby vhodil mince a zaplatil vybraný nápoj.
3. Zákazník vhodí minci do automatu.
4. Automat vyhodnotí vhozený obnos.
5. DOKUD je hodnota vhozeného obnosu menší než cena nápoje, opakuje se krok 2.
6. JE-LI hodnota vhozeného obnosu větší než cena nápoje, PAK automat vypočte přeplatek, a připraví přeplatek na odebrání.

Alternativní scénáře:
Zákazník nebyl schopen nápoj zaplatit – StornovatNakup.

Obrázek 4: Ukázka scénáře pro případ užití ZaplatitNapoj.

4.2 Doporučený postup tvorby scénáře

Nejprve si vytvoříme prázdnou šablonu, do které budeme postupně vyplňovat jednotlivé položky. Se správným určením aktérů, kroků scénáře a podmínek nám mohou pomoci návodné otázky. Po jejich zodpovězení můžeme specifikovat podmínky i jednotlivé kroky scénáře.

1. Určení hlavního aktéra: Kdo spouští případ užití?
2. Stručný popis případu užití: Co vyžaduje aktér po tomto případu užití?
3. Vstupní podmínky: Co musí být splněno, aby případ užití začal?

4. Následné podmínky: Až případ užití skončí, jaké změny v systému nastaly?

5. Popis sekvencí scénáře: Jak systém reaguje během jednotlivých kroků realizujících případ užití?

6. Určení vedlejších aktérů: Objevil se v průběhu popisu realizace případu užití jiný aktér?

7. Kontrola konzistence předchozí tvorby modelu: Nebyl popisem porušen soulad aktérů z diagramu případů užití a scénáře?

Odpovědi na otázky přiřadíme do patřičných kolonek šablony popisu scénáře.

4.3 Úkoly

Vytváření scénáře

Nejprve si před vytvořením scénáře případu užití ZvolitNapoj postupně odpovíme na otázky.

1. Kdo spouští případ užití? – Zakaznik

2. Co vyžaduje aktér po tomto případu užití? – Zakaznik si přečte nabídku nápojů a zvolí si nápoj.

3. Co musí být splněno, aby případ užití začal? – Nápojový automat musí být funkční.

4. Až případ užití skončí, jaké změny v systému nastaly? – Nápojový automat přijal požadavek Zakaznika.

5. Jak systém reaguje během jednotlivých kroků realizujících případ užití? – Systém nabídne typy nápojů. Systém přijme volbu Zakaznika.

6. Objevil se v průběhu popisu realizace případu užití jiný aktér? – Žádný další aktér nevstoupil do případu užití.

7. Nebyl popisem porušen soulad aktérů z diagramu případů užití a scénáře? – K žádné změně ve vztahu mezi aktérem a případem užití nedošlo.

Do vytvořené šablony zapíšeme odpovědi a postupně rozepíšeme jednotlivé kroky.

Samostatná práce

Vytvořte scénář pro některý z případů užití systému Taxis a převeďte jej cvičícímu.

UC: ZvolitNápoj
ID: UC1
Stručný popis: Zákazník si vybere nápoj z nabídky nápojů.
Hlavní aktéři: Zákazník
Vedlejší aktéři: Nejsou
Vstupní podmínky: Automat je funkční, je k dispozici nabídka nápojů.
Následné podmínky: Automat je ve stavu přijetí požadavku Zakaznika.
Hlavní scénář: 1. Zákazník přistupuje k automatu. 2. Automat zobrazuje nabídku nápojů. 3. Zákazník si vybere nápoj dle nabídky a zadá svou volbu automatu. 4. Automat potvrdí volbu.
Alternativní scénáře: Vybraný nápoj je vyprodán.

Obrázek 5: Jednoduchý scénář pro případ užití ZvolitNapoj.

5 Tvorba třídního diagramu

Cíl cvičení

Vyhledat třídy, jejich atributy a operace. Navrhnout vazby mezi třídami. Vytvořit třídní diagram.

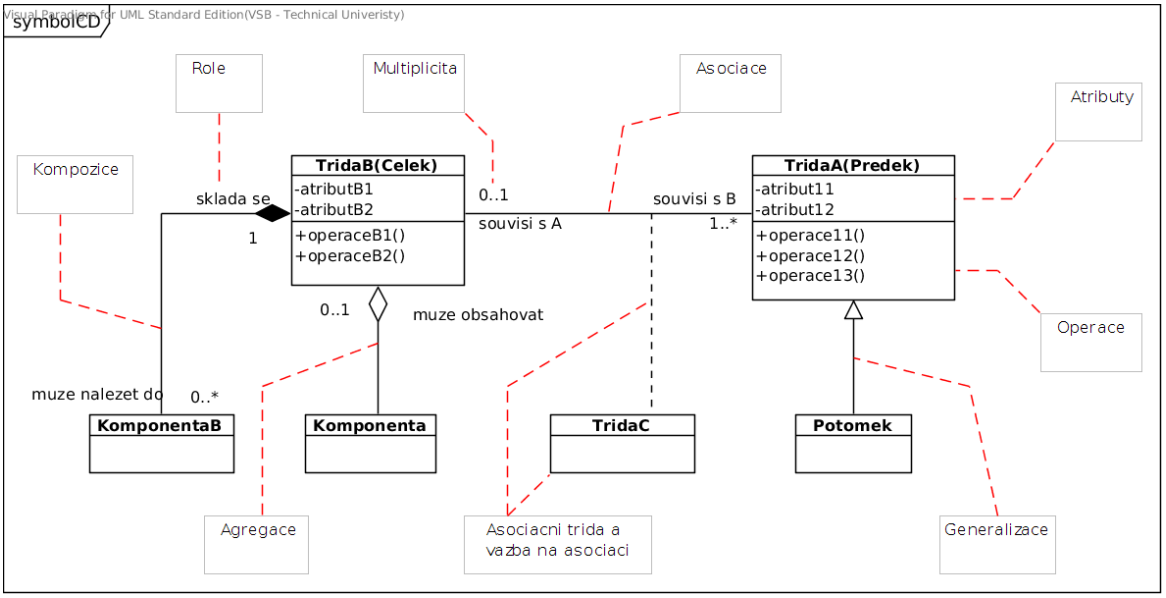
5.1 Teoretický základ

Diagram tříd zachycuje strukturu systému, patří mezi statické diagramy. Znázorňuje třídy a jejich vztahy.

Třídu chápeme jako kategorii nebo skupinu věcí, které mají podobné vlastnosti (atributy) a vykazují stejné nebo podobné chování (operace/metody).

5.1.1 Syntaxe diagramu tříd

Třídní diagram lze vnímat jako graf složený z uzlů a hran. Uzly jsou reprezentovány třídami a hrany představují vztahy mezi nimi. Používané symboly a jejich význam jsou zachyceny na obrázku 6.



Obrázek 6: Obrázek symbolů aktivitního diagramu (vytvořeno v akademické licenci Visual Paradigm).

Třída se zachycuje pomocí obdélníku, který může být rozdělen do tří částí.

- Název třídy (povinná část) má korespondovat s jejím účelem,
- atributy (nepovinná část) charakterizují vlastnosti třídy (objektů),

- operace (nepovinná část), kde každá operace specifikuje změnu stavu cílového objektu nebo dotaz (ten vrací návratovou hodnotu volajícímu objektu). Operace specifikuje výsledek chování volaného objektu, chování jako takové je specifikováno metodou (kódem).

Atributy jsou popsány svým názvem, dále může být uveden datový typ a viditelnost. U operací uvádíme název, případně argumenty operace a návratové typy pro lepší pochopení modelované struktury.

Vztahy mezi třídami jsou specifikovány několika způsoby, podle charakteru vztahu:

- asociace – obecný vztah mezi třídami, specifikuje vazbu mezi jejich instancemi (objekty tříd propojených asociací si mohou posílat zprávy),
- agregace – asociace, která zachycuje vztah celek – část, komponenta může existovat i bez celku, případně se může stát součástí jiného celku,
- kompozice – silnější vazba než agregace - zrušením celku automaticky zaniká i komponenta, komponenta patří právě do jednoho celku,
- dědičnost (generalizace) – hierarchický vztah tříd, kdy potomek (subclass, podtyp) dědí atributy a operace svého předka (superclass, nadtyp), potomek může mít navíc proti zděděným atributům a operacím své vlastní, zděděné pak mohou u potomka modifikovány,
- závislost – vztah mezi třídami (obecně ovšem mezi dvěma prvky modelu systému), kdy změna jedné třídy (nezávislé) ovlivní druhou třídu (závislou),
- asociační třída – třída, která má vztah nikoli k jiné konkrétní třídě, ale k asociaci mezi třídami.

Vztahy navíc upřesňujeme pomocí multiplicity (kardinality).

Zápis	Význam
0..*	0 a více
*	0 a více
0..1	0 nebo 1
1..*	1 a více
1..1	právě 1
8..11	8 a 11
4,6,12,20	4,6,12 nebo 20
7..*	7 a více

Vyhledávání analytických tříd

K vyhledání tříd je možné použít různé postupy, jako jsou například:

- návrh tříd na základě znalosti domény,
- uspořádat interview se zadavatelem a uživateli a zaznamenat jejich názory,

- pomocí textové analýzy z textových podkladů nalézt kandidáty na třídy, atributy a operace,
- metodou brainstormingu analytického týmu a doménových expertů vytvořit karty CRC (Class-Responsibility-Collaboration, třída-zodpovědnost-spolupráce).

5.2 Doporučený postup

Použijeme postup, který s pomocí textové analýzy hledá kandidáty tříd, atributů a jejich operace z pohledu zodpovědnosti každé třídy.

1. Textovou specifikaci systému procházíme a slova splňující vlastnost kandidáta na třídu, atribut či operaci označíme barevně.
2. Sloučíme k sobě pojmy, které mají stejný význam (synonyma).
3. Kandidáty setřídíme a vytvoříme sadu pojmů, které spolu vytvářejí kostru třídy.
4. K prvotnímu návrhu doplníme atributy či operace, které vyplývají z předešlých modelovacích činností (UC, scénáře, ...).
5. Přeneseme elementy z předchozí práce do grafického editoru a vytvoříme první verzi třídního diagramu.

5.3 Úkoly

Vytváření diagramu

1. V textovém editoru budeme zpracovávat specifikaci systému – postupně procházíme text a slova splňující vlastnost kandidáta označíme barevně.

   Podstatné jméno jako kandidát na třídu – červeně.
   Podstatné jméno jako kandidát na atribut – modře.
   Sloveso jako kandidát na operaci – zeleně.

Nápojový automat je naplněn různými druhy nápojů. Jejich nabídka je viditelná pro zákazníka ve formě označení nápoje, druhu nápoje a ceny. Zákazník přistupuje k nápojovému automatu, vybere si nápoj z nabídky. Aby mu mohl být nápoj vydán, je vyzván k vhazování mincí do otvoru pro mince. Po vyhodnocení dostatečného obnosu nápojový automat vhodí do výdejního boxu pro nápoj zaplacený nápoj. V případě přeplatku se do boxu na mince vrátí přeplatek.

Obrázek 7: Ukázka postupu textové analýzy.

2. Synonyma v této stručné specifikaci nejsou.

3. Sloučíme k sobě nalezené významy, které znamenají kandidáta na třídu, případně i s jeho atributy a operacemi vytvoříme základ pro třídu.

   Třída: Nabídka
   Atributy:
   Operace: zobrazitNabidku

   Třída: Napoj
   Atributy: oznaceniNapoje, druhNapoje, cena
   Operace:

4. Doplníme kandidáty o další atributy či operace. Z předešlé dokumentace doplníme další adepty na třídy, včetně dříve navrhnutých odpovědností.

   Třída: Nabídka
   Atributy:
   Operace: zobrazitNabidku

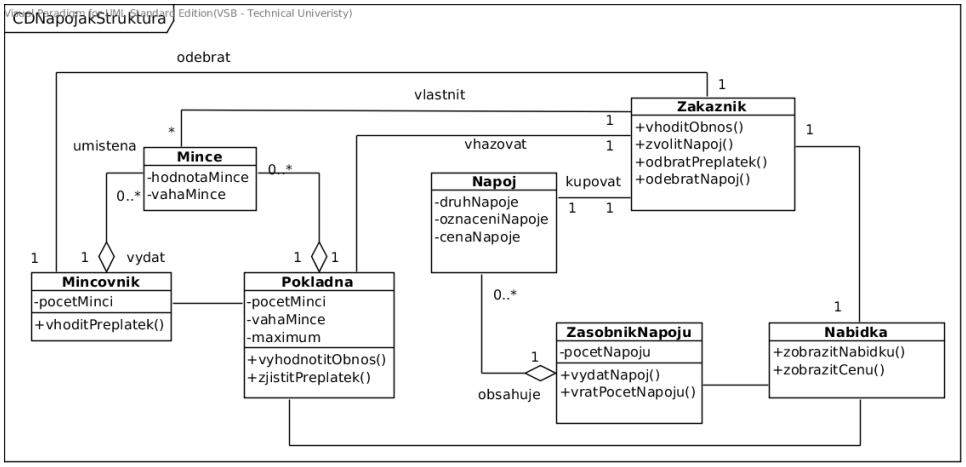
   Třída: Napoj
   Atributy: oznaceniNapoje, druhNapoje, cenaNapoje
   Operace: vratCenu
   Spoluprace: ZasobnikNapoju

   Třída: ZasobnikNapoju
   Atributy: pocetNapoju
   Operace: jeDruhKDispozici, vhoditNapojVydejniku
   Spoluprace: Napoj, Nabidka

   Třída: Zakaznik
   Atributy:
   Operace: zvolitNapoj, vhoditMinci, odebratNapoj, odebratPreplatek
   Spoluprace: Napoj, Nabidka, Mince

5. Spustíme grafický editor, zvolíme tvorbu třídního diagramu. Vytvoříme třídy, které jsme nadefinovali.
6. Mezi třídami vytvoříme vazby a doplníme multiplicitu, případně označíme vazby pomocí názvů vztahů či pojmenováním rolí prvků.

Podrobný popis tvorby třídního diagramu v prostředí Visual Paradigm je umístěn na webových stránkách `http://www.visual-paradigm.com/support/documents/vpuserguide/94/2576/7190_creatingclas.html`.



Obrázek 8: Třídní diagram pro nápojový automat – ukázka jednoho z možných řešení struktury systému.

Samostatná práce

1. Dotvořte třídní diagram pro Nápojový automat.
   - Doplňte do první verze diagramu třídy pro ProvozovateleAutomatu, Zasobovace(DoplnovaceNapoju), Pokladnika. Bude mezi nimi specifická vazba?
   - Z dokumentace převezměte další adepty na třídy a doplňte je do třídního diagramu včetně vztahů.
2. Vytvořte třídní diagram pro systém Taxis.

6 Tvorba sekvenčního diagramu

Cíl cvičení

Pro vybrané případy užití a jejich scénáře detailně zachytit komunikaci mezi objekty.

6.1 Teoretický základ

Sekvenční diagram patří do skupiny diagramů interakcí. Ukazuje, jak objekty komunikují navzájem mezi sebou v časové rovině. Představuje podrobný popis, jak bude realizován určitý scénář, případně návazné scénáře. Jeho smyslem je zachytit postupné zasílání zpráv mezi objekty, zobrazuje tak činnosti několika účastníků během modelované interakce. Představuje spolupráci objektů, která vede ke splnění určité úlohy při užití systému.

Prvky sekvenčního diagramu jsou objekty nebo aktéři (účastníci), jejich čáry života a zprávy, které si mezi sebou posílají.

Čára života reprezentuje účastníka v interakci s ostatními. Označuje se jako buď jako reprezentant třídy - anonymní objekt, případně jako konkrétní instance třídy.

Aktivita účastníka představuje časový interval, kdy je daný účastník aktivní, kdy provádí nějakou činnost, včetně čekání na návrat ze zavolané operace. Překrývá čáru života po dobu aktivity účastníka.

Zpráva zobrazuje komunikaci mezi účastníky. Je reprezentována šipkou od jedné čáry života účastníka k další. Její směr představuje od koho (odesílatel) komu (příjemce) je zpráva zaslána. Umístění zpráv shora dolů mezi čárami života postupně vytváří sekvenci zpráv a znázorňuje tak plynutí času shora dolů. Zobrazení návratové zprávy v případech, kdy je zřetelný smysl, není nutno uvádět.

Pro naše potřeby vystačíme s několika základními typy zpráv.

- *Synchronní zpráva* – odesílatel zprávy čeká na odpověď, teprve poté může provádět další akce.
- *Asynchronní zpráva* – odesílatel zprávy nečeká na odpověď, může po jejím odeslání začít s dalšími akcemi.
- *Rekurzivní zpráva* – odesílatel zasílá zprávu sám sobě.

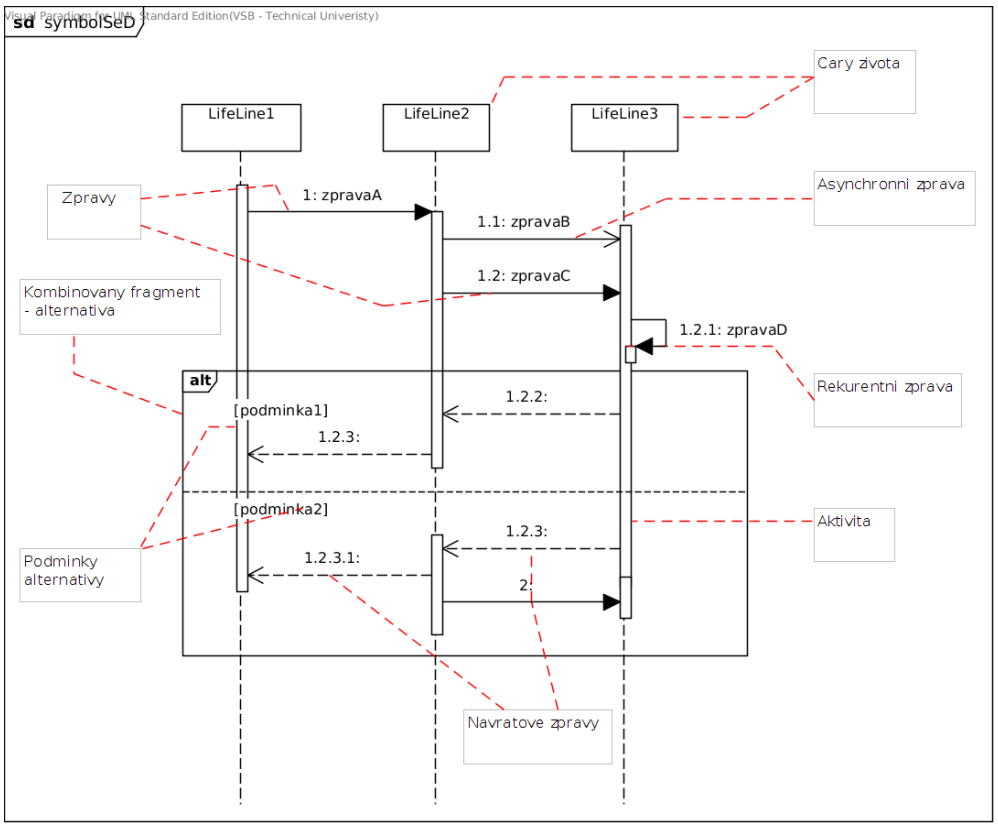
Parametry zpráv, předávaných mezi účastníky, není povinné zapisovat, ale mohou pomoci v při pochopení interakce.

Fragment zahrnuje určitou část interakcí a má speciální význam. Ve většině případů se dělí na více částí, realizece určité části je podmíněna splněním podmínky pro interakce zde definované. Uvádíme několik základních typů.

- *Alt – Alternativa* je fragmentem rozděleným do minimálně dvou částí, kdy interakce proběhnout v té části, kde je splněna podmínka.

- *Opt – podmíněný fragment* proběhne jen při splnění podmínky (de facto jednosložková alternativa).
- *Loop – smyčka* má definovánu podmínku pro opakovaný průběh obsahu fragmentu.

Zobrazení prvků UML v sekvenčním diagramu je nakresleno na obrázku 18.



Obrázek 9: Obrázek se symboly sekvenčního diagramu (vytvořeno v akademické licenci Visual Paradigm).

6.2 Doporučený postup

Při vytváření sekvenčního diagramu se doporučuje postupovat v následujících krocích.

1. Vybrat si scénář, který bude detailněji rozebrán a definovat zprávy mezi objekty.
2. Rozeznané objekty zaznamenat do sekvenčního diagramu.
3. Podle scénáře doplňovat zprávy podle logické a časové posloupnosti.

4. Kontrolovat konzistenci s třídním diagramem. V případě nesrovnalostí upravit.

5. Upravit zprávy podle typu.

6.3 Úkoly

Vytvoření sekvenčního diagramu

Pro vytváření sekvencí zpráv vezmeme jako základ scénář a z třídního diagramu vybereme ty části struktury, které se účastní realizace scénáře.

1. Vybereme si scénář, který budeme kreslit ve formě sekvenčího diagramu – *Zaplatit-Napoj*.

2. V grafickém editoru zvolíme nový sekvenční diagram, nazveme jej SeDZaplatit-Napoj a zaznamenáme do něj objekty účastnící se interakcí. (Zakaznik, Nabidka, Pokladna, Mincovnik).

3. Podle scénáře Zakaznik vyžaduje informaci o ceně vybraného nápoje. Odešle tedy zprávu adresovanou Nabidce *zobrazitCenu*. Návratovou hodnotou bude cena vybraného nápoje.

4. Postupně budeme doplňovat zprávy podle logické a časové posloupnosti, vycházíme z nabídky, kterou nám prostředí editoru nabízí u jednotlivých čar života, případně doplníme vlastní názvy zpráv. Pak je ale nutno zajistit konzitenci u tříd v diagramu tříd.

5. Doplníme opakování pro vhazování obnosu, podmínkou je nedostatečná hodnota vhozená Zakaznikem. Podmínku vepíšeme do formuláře, který umožňuje upravovat operand fragmentu *loop*.

6. Doplníme fragment *opt* pro případ, kdy Zakaznik přeplatil a bude mu vrácen přeplatek. Podmínku *hodnotaObnosu>cenaNapoje* umístíme jako omezení operandu fragmentu.

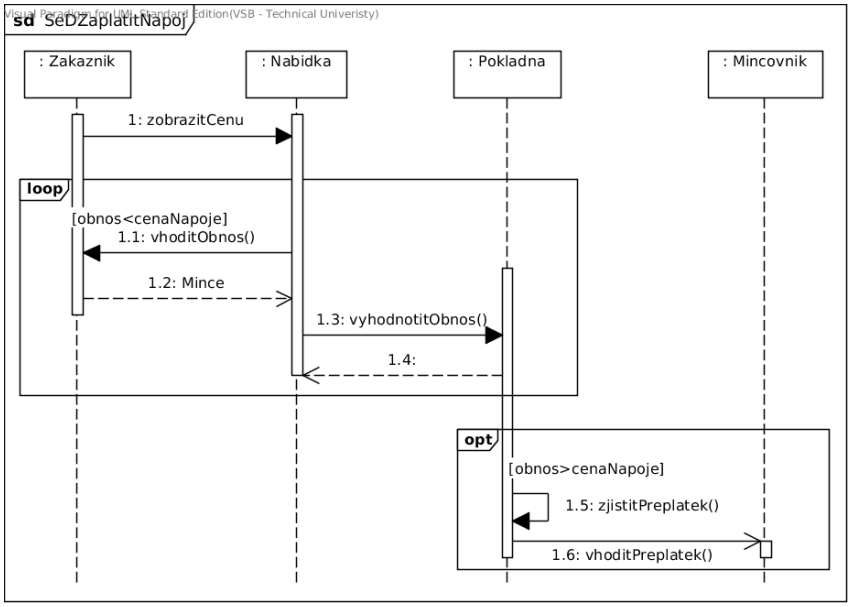
7. Pro zprávu *vhoditPreplatek* upravíme její typ na asynchronní.

Výsledke postupného vytváření sekvenčního diagramu je předveden na obrázku 10.

Samostatná práce

1. Vytvořte sekvenční diagram pro případ užití *ZvolitNapoj*.

2. Vytvořte sekvenční diagram pro vytvořené scénáře systému Taxis.



Obrázek 10: Sekvenční diagramu pro případ užití ZaplatitNapoj (vytvořeno v akademické licenci Visual Paradigm).

7 Konzistence modelu

Cíl cvičení

Naučit se kontrolovat konzistenci modelu.

7.1 Teoretický základ

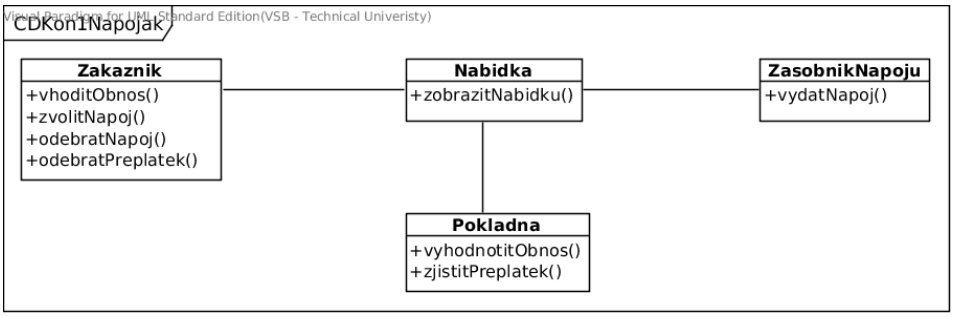
Při modelování systému je důležité udržovat jednotlivé diagramy konzistentní. Napomáhá tomu iterační postup při tvorbě modelu a vědomí souvislostí mezi modely. Existuje celá řada pravidel, která konzistenci napomáhají udržet.

Uvedeme základní, která platí mezi diagramy tříd a sekvenčním.

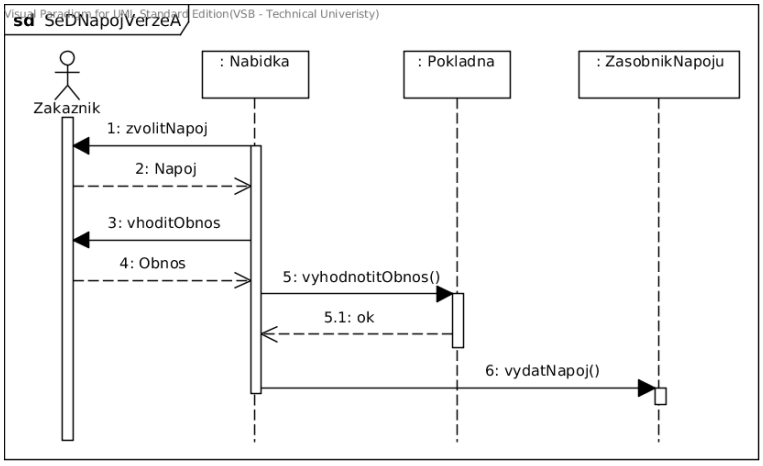
- Signatura operací tříd, použitých v sekvenčním diagramu, musí odpovídat signaturám zpráv posílaných mezi objekty.
- Při směřování zpráv je vhodné kontrolovat, zda je příjemcem objekt, u nějž bude spouštěna požadovaná metoda.
- Pro udržení konzistence obou diagramů je nuntá existence operací u třídy, kam je zasílána zpráva v sekvenčním diagramu.
- Existenci všech zpráv v sekvenčním diagramu, která odpovídá operaci v třídním diagramu, je možné ověřit až v okamžiku, kdy jsou všechny sekvenční diagramy hotovy. Pak by neměly existovat operace/metody, jejichž spuštění není v celém modelu vůbec vyžadováno.

7.2 Úkoly

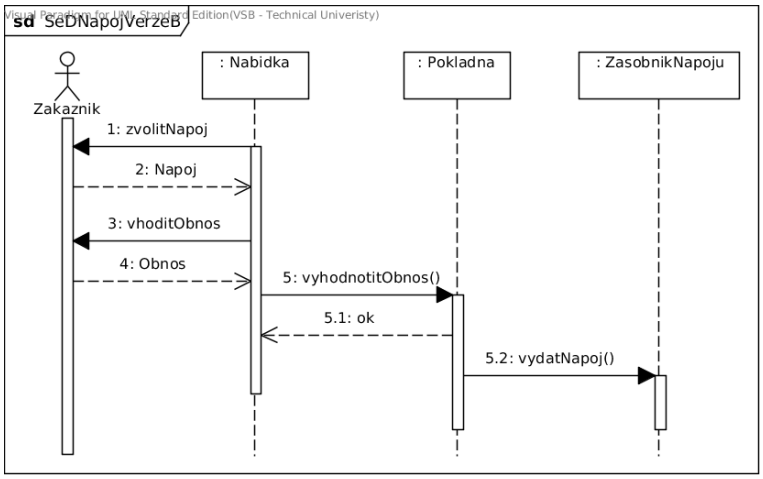
- Zkontrolujte, zda je dodržena konzistence třídního a sekvenčního diagramu ZaplatitNapoj, ZvolitNapoj podle uvedených pravidel.
- Ověřte konzistenci na následujících diagramech. Na obrázku 11 je zakreslena část struktury nápojového automatu pomocí třídního diagramu. Vyberte sekvenční diagram z možností A, B, C, D takový diagram, který konzistenci dodržuje.
- Uveďte a pojmenujte chyby v ostatních případech.



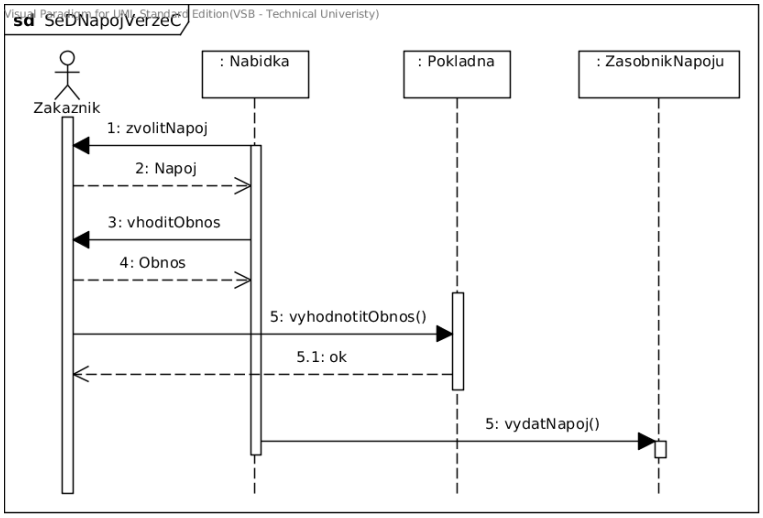
Obrázek 11: Část třídního diagramu nápojového automatu.



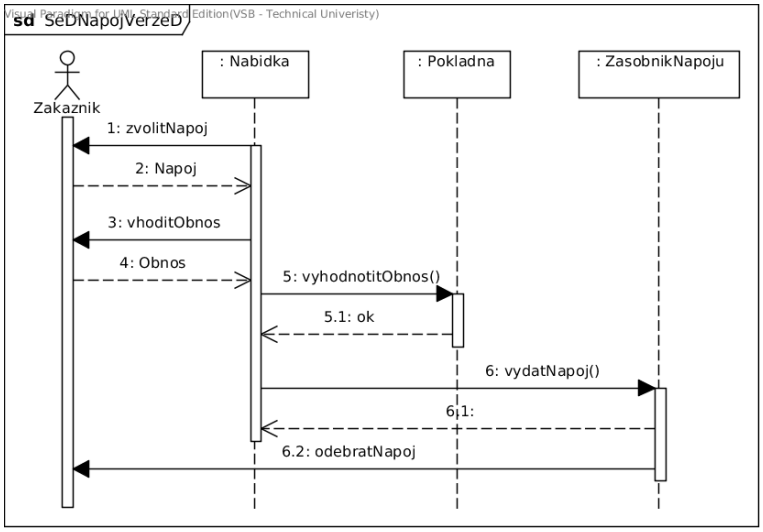
Obrázek 12: Sekvenční diagram, varianta A.



Obrázek 13: Sekvenční diagram, varianta B.



Obrázek 14: Sekvenční diagram, varianta C.



Obrázek 15: Sekvenční diagram, varianta D.

8 Tvorba diagramu aktivit

Cíl cvičení

Vytvořit posloupnost akcí, jejichž provedení modeluje vybranou aktivitu.

8.1 Teoretický základ

Diagram aktivit patří mezi diagramy popisující chování systému či jeho částí. Používáme jej pro modelování procedurální logiky, procesů a zachycení workflow [2]. Každý proces, který se v systému odehrává, popisujeme jako sadu činností (akcí) a rozhodnutí, které řídí, jak tyto činnosti půjdou za sebou. V diagramu aktivit modelujeme proces pomocí *uzlů* a *hran*.
Uzly mohou mít trojí charakter:

- akční – atomická činnost, v rámci aktivity nedělitelná. Lze ovšem zakreslovat i vnořené aktivity formou akčního uzlu, který reprezentuje aktivitu detailně reprezentovanou dalším diagramem aktivit.
- objektový – zastupuje objekt použitý v rámci dotyčné aktivity,
- řídící – řídí cestu uvnitř aktivity.

Hrany, které znázorňují tok v rámci aktivity, dělíme na:

- řídící – řídí cestu uvnitř aktivity,
- objektové – zastupují cestu objektů v rámci aktivity.

Posloupnost jednotlivých prvků v diagramu aktivit určuje pořadí, v jakém budou činnosti vykonávány. Pořadí je znázorněno přechodovými hranami. Zodpovědnost za konkrétní činnost je určena umístěním uzlu do *plavecké dráhy*, která je přiřazena aktérům či objektům.

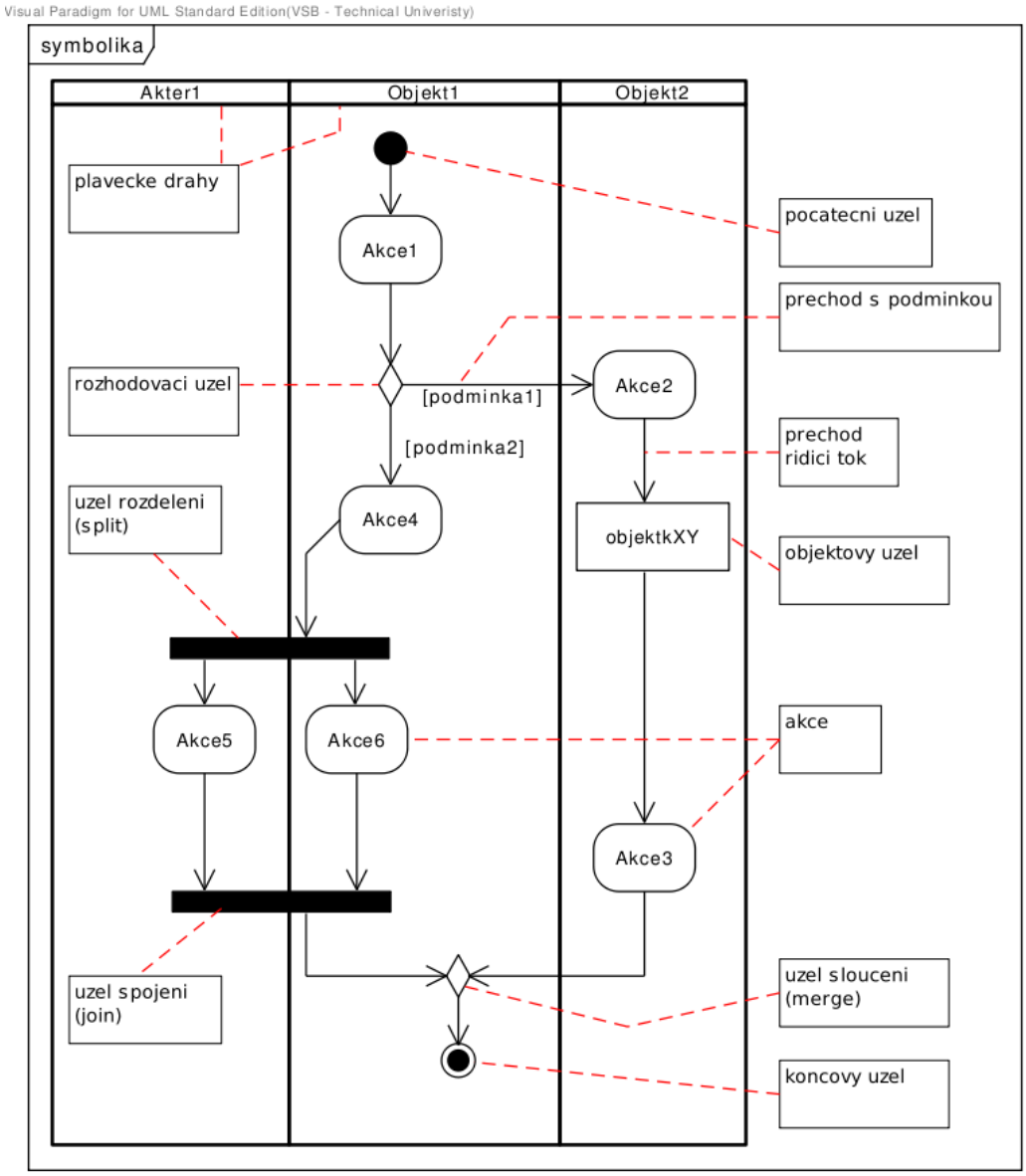
8.1.1 Syntaxe diagramu aktivit

Pužívané prvky UML a jejich význam v aktivitn9m diagramu jsou popsány v následujícím přehledu a zobrazeny na obrázku 16.

Počáteční uzel představuje bod, kde aktivita začíná.

Koncový uzel označuje konec aktivity. Aktivita může končit v různých větvích, koncových uzlů může být více.

Akční uzel představuje dále nedělitelný element chování objektu. Jeho název charakterizuje obsah činnosti, kterou uzel symbolizuje. Může jít o zavolání operace jiné třídy, volání funkce, zaslání signálu, nastavení atributu, zjištění hodnoty atributu apod.



Obrázek 16: Obrázek symbolů aktivitního diagramu (vytvořeno v akademické licenci Visual Paradigm).

Řídící tok ukazuje souslednost jednotlivých akcí, jedná se o tok řízení celé aktivity.

Rozhodovací uzel (rozvětvení, fork) umožňuje rozdělení toku řízení do dvou či více větví na základě podmínek přechodu. Podmínky musí být vzájemně ve vztahu vylučovacím.

Slučovací uzel (merge) představuje sloučení větví do jedné. Má stejný symbol jako uzel rozhodovací, jen do něj směřují slučované přechody. Vychází z něj pouze jeden přechod.

Uzel pro souběh činností (paralelní větve, rozdělení, split) je reprezentován jedním vstupem a několika výstupními přechody. Představuje souběžné provádění akcí na řídících větvích nezávisle za sobě a bez určování pořadí.

Uzel spojení (join) má stejný symbol jako uzel paralelismu, směřují však do něj všechny souběžné přechody a vychází pouze jeden přechod. Ten je možný teprve po ukončení všech akcí v paralelních větvích.

Plavecká dráha ukazuje zodpovědnost účastníků aktivity. Každá dráha je označena názvem role nebo objektu a akce se do ní přiřazují na základě zodpovědnosti za prováděné akce. Předávání řízení může probíhat i mezi dráhami, ukazuje se tím předávání zodpovědnosti za provádění akcí mezi účastníky aktivity.

8.2 Doporučený postup tvorby diagramu

Při vytváření aktivitního diagramu budeme postupně řešit tyto kroky:

1. Vybrat si úroveň detailu, se kterým budeme přistupovat k formulování modelované aktivity.
2. Pro zvolený případ naformulovat hlavní akce, které případ popisují.
3. Od počátečního uzlu pomocí řídících hran propojit akce mezi sebou podle pořadí, jak budou uskutečňovány.
4. Na potřebná místa doplnit rozhodovací uzly, doplnit podmínky pro předávání řízení. Doplnit paralelní větvení.
5. Do diagramu zakreslit plavecké dráhy, zařadit akce dle zodpovědnosti zúčastněných aktérů či objektů.

8.3 Úkoly

Vytváření diagramu

Realizujeme kreslením na tabuli či papíře, nebo použijeme software pro tvorbu diagramů.

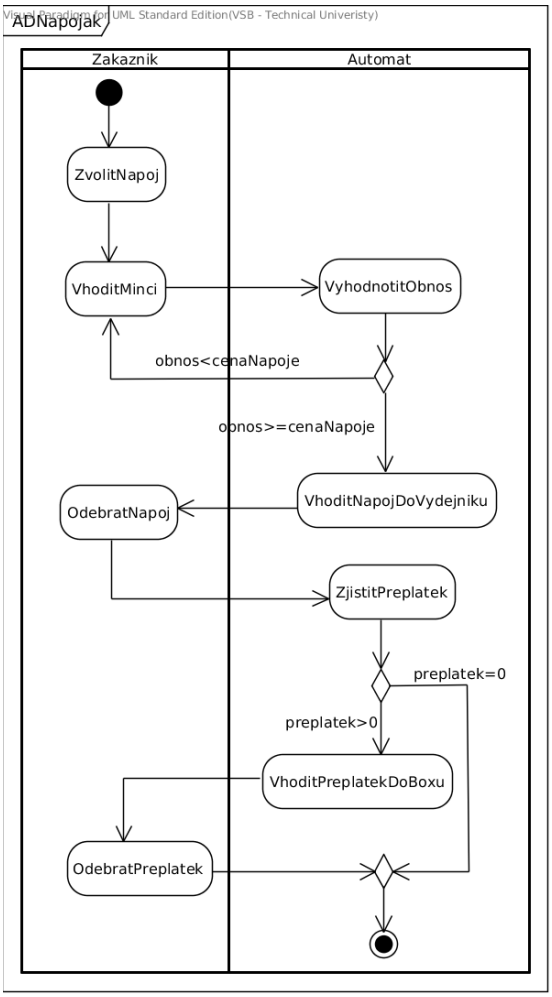
1. Zvolíme tvorbu aktivitního diagramu jako celkového pohledu na proces koupě nápoje z automatu.
2. Začneme vyhledávat akce, které celou aktivitu zachytí a zapíšeme si jejich seznam.
   - ZvolitNapoj,
   - VhoditMinci,

- OveritDostupnostNapoje,
- OdebratNapoj,
- OdebratPreplatek,
- VyhodnotitHodnotuMince,
- ...

3. Nejprve zakreslíme počáteční uzel.
4. Vytvoříme všechny akční uzly a zakreslíme hrany mezi nimi podle předávání řízení.
5. Doplníme rozhodovací uzly. Například: po vhození mince následuje rozhodování, zda mince má dostatečnou hodnotu a je možné vydat pokyn pro vhození nápoje do boxu pro odebrání nebo je nutné přenést zodpovědnost na Zakaznika a vyžádat od něj další minci. Vytvoříme rozhodovací uzel a hrany s podmínkami [obnos=>cenaNapoje] a [obnos<cenaNapoje] dokreslíme tak, aby byly ošetřeny obě dvě možnosti.
6. Doplníme další větve s podmínkou pro zjišťování přeplatku..
7. Do diagramu zaneseme plavecké dráhy a přiřadíme do nich akce dle zodpovědnosti Zakaznika a Automatu.

Samostatná práce

1. Vytvořte aktivitní diagram pro systém Taxis, který bude popisovat činnosti Zakaznika a Ridice od zahájení jízdy až po její ukončení.



Obrázek 17: Aktivitní diagram zachycující logiku fungování nápojového automatu (vytvořeno v akademické licenci Visual Paradigm).

9 Tvorba stavového diagramu

Cíl cvičení

Nalézt stavy důležitých objektů systému, provázat je přechody a vytvořit stavový diagram.

9.1 Teoretický základ

Stavový diagram patří mezi diagramy chování, popisuje životní cyklus vybraného objektu, komponenty, případně celého systému chápaného jako objekt. UML jej definuje jako deterministický automat, což znamená, že pro každý stav musíme nadefinovat podmínky, při kterých automat přejde do stavu jiného. Pokud z jednoho stavu může vést více přechodů, každý z nich je určen jinou podmínkou. Stavový diagram popisuje dynamické vlastnosti objektů systému, reprezentuje všechny stavy, do kterých se popisovaný objekt může dostat během svého životního cyklu. Lze díky němu pochopit chování objektů v různých okamžicích provozu systému.

UML definuje stavový diagram pomocí dále popsaných prvků.

Stav vyjadřuje situaci, kdy objekt splňuje nějakou podmínku, provádí nějakou akci nebo čeká na událost. Znázorňujeme jej obdélníkem s oblými rohy. Do obdélníku lze napsat aktivity, které se v daném stavu provádí (mohly být zaznamenány i diagramem aktivit pro daný stav). Například po dosažení věku 18 let, je každý občan označen jako plnoletý, může provádět různé úřední úkony, oženit se/vdát se, volit, ziskatRidicak, atd. a toho plynou možnosti jiných stavů občana.

U stavu lze rozlišit aktivitu, která probíhá během trvání stavu. Rozepisujeme pak vstupní, výstupní a interní akce (entry/*vstupniAkce*, exit/*vystupniAkce*, do/*akce*).

Přechod mezi stavy představuje spojení mezi dvěma stavy a je vyjádřen orientovanou hranou. Objekt přejde z jednoho stavu do druhého stavu za splnění určitých podmínek. Přechod také může vést z jednoho stavu do téhož stavu.

Událost, která způsobí přechod, chápeme jako něco, co se stane v určitém časovém okamžiku, neurčujeme její trvání. Je znázorněna jako doplňující informace u šipky přechodu.

Podmínky přechodů Podmínka/y se uvádí v hranatých závorkách přímo u šipky přechodu: [podmínka].

Akce, které se během přechodu konají (například výpočet) určují realizaci přechodu.

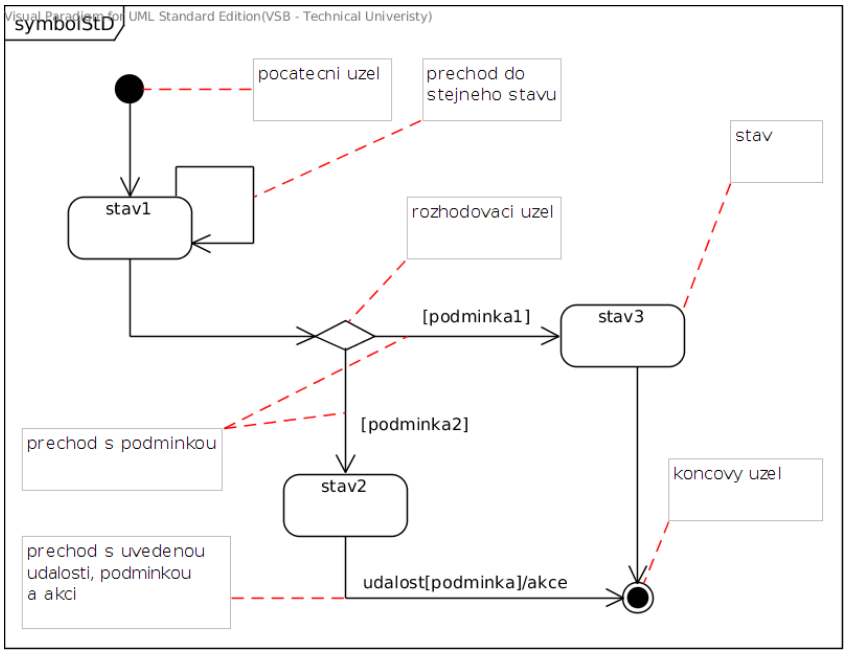
Součástí stavového diagramu je i několik pseudostavů, které jsou chápány jako stavy se speciálním chováním.

Počáteční bod je v diagramu pouze jeden. Odtud diagram začíná. Obvykle se z něj přechází bez podmínky k dalšímu stavu, ve kterém se na začátku ocitneme. Je-li podmínka uvedena, musí být automaticky splněna. Z počátku je možný pouze jeden přechod.

Koncový bod je v diagramu obvykle jeden, ale podle UML jich může být i více. Je vhodné rozložit tok digramu od levého horního rohu k pravému dolnímu, tj. počáteční bod je nahoře, koncový bod dole.

Volba pomáhá rozdělit přechod do více segmentů. Po vyhodnocení volby se přechází do stavu, u jehož segmentu(přechodu) byla splněna strážní podmínka.

Symbolika prvků stavového diagramu je uvedena na obrázku *symbolStD*.



Obrázek 18: Obrázek se symboly stavového diagramu (vytvořeno v akademické licenci Visual Paradigm).

9.2 Doporučený postup tvorby diagramu

Při tvorbě stavového diagramu je vhodné postupovat v následujících krocích.

1. Nejprve nalézt významné objekty, ke každému se bude tvořit jeden diagram.
2. Definovat jeho stavy pomocí analýzy hodnot atributů.
3. Zaznamenat stavy do diagramu.
4. Dopsat popis stavu do symbolu stavu.
5. Nalézt přechody mezi stavy, případně přechody s návratem do stejného stavu.
6. Zaznamenat přechody do diagramu.

7. Nalézt omezení u jednotlivých přechodů.

8. Zaznamenat omezení do diagramu.

9.3 Úkoly

Vytváření diagramu

Realizujeme kreslením na tabuli či papíře, nebo použijeme software pro tvorbu diagramů.

1. Za významný objekt modelu nápojového automatu zvolíme Pokladnu.

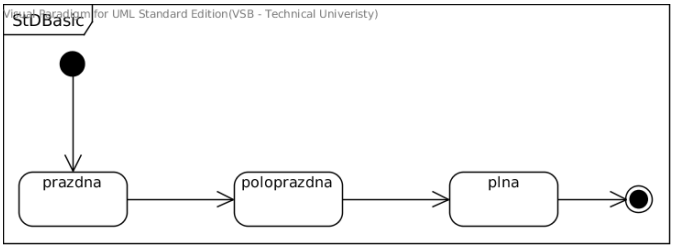
2. Nadefinujeme stavy Pokladny. Určujícím atributem je počet přijatých mincí. Doména tohoto atributu je od 0 po *maximum* (dáno velikostí pokladny). Je-li počet mincí v pokladně = 0, označíme stav Pokladny *prazdna*.
   Je-li počet mincí v pokladně > 0, označíme stav Pokladny *poloprazdna*.
   Je-li počet mincí v pokladně = *maximum*, označíme stav Pokladny *plna*.

3. Pro každý nalezený stav vytvoříme symbol s odpovídajícím názvem a umístíme do diagramu.

4. Vytvoříme počáteční bod (v některých nástrojích může při založení nového diagramu již být na kreslící ploše) a z něj zvolíme přechod do stavu *prazdna*.

5. Vyhledáme možné přechody mezi stavy. Ze stavu *prazdna* po přijetí mince dojde ke změně stavu *poloprazdna*. Do stavu *plna* lze přejít pouze vhozením poslední možné mince ze stavu *poloprazdna*.

   Výsledek záznamu základních přechodů je patrný na obrázku 19.

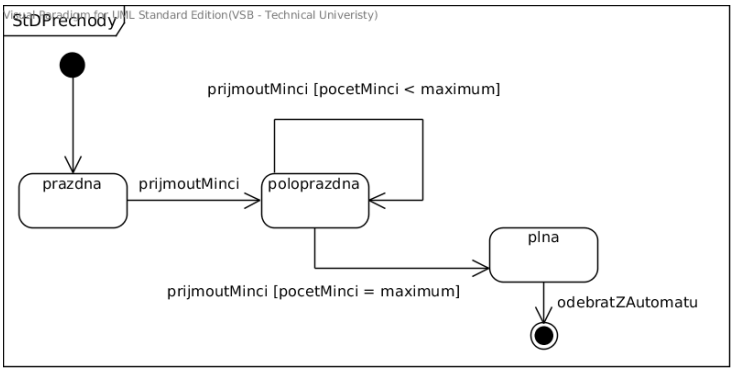


Obrázek 19: Obrázek základu stavového diagramu (vytvořeno v akademické licenci Visual Paradigm).

6. Dále zde máme přechod s návratem do stejného stavu. Jedná se o případ, kdy Pokladna již obsahuje minci – je *poloprazdna*, další minci přijímá a přijetí nepovede k úplnému zaplnění Pokladny, takže se vrátí zpět do stavu *poloprazdna*.

7. Nadefinujeme omezení u jednotlivých přechodů. U stavu *poloprazdna* mohou nastat dva přechody, protože na základě naplněnosti Pokladny může dojít k přechodu do stejného stavu nebo do stavu *plna*.

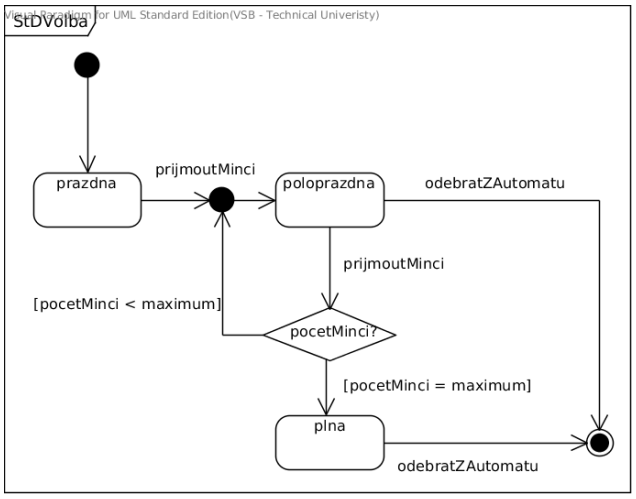
8. Zaznamenáme události a omezení přechodů do diagramu (viz obrázek 20).



Obrázek 20: Obrázek stavového diagramu s událostmi a podmínkami přechodů (vytvořeno v akademické licenci Visual Paradigm).

9. Vložíme rozhodovací uzel a rozložíme přechod *prijmoutMinci* s podmínkami tak, aby do rozhodovacího uzlu vcházel přechod s událostí. Po vyhodnocení počtu mincí z rozhodovacího uzlu vychází dva segmenty, každý dle strážní podmínky míří do stavu, který je dán ne/splněním podmínky $pocetMinci < maximum$, viz obrázek 21.

10. Upravíme přechody do koncového bodu ze stavů *poloprazdna* a *plna*. Ukončení životnosti Pokladny ve stavu *prazdna* nepředpokládáme.



Obrázek 21: Obrázek po dokončení stavového diagramu pro objekt Pokladna (vytvořeno v akademické licenci Visual Paradigm).

Samostatná práce

1. Vytvořte stavový diagram pro jiný objekt Nápojového automatu.
2. Vytvořte stavový diagram pro některý objekt systému Taxis.
3. Vytvořte stavový diagram pro objekt Objednavka, která má definovány tyto možné stavy: *nova*, *doplnenaOInformace*, *dokoncena*, *zrusena*, *zaplacena*. Znáte několik událostí a podmínek přechodů. Po vytvoření objednávky v systému je nutno ji doplnit o další upřesňující informace. Dále je odeslána zákazníkovi žádost o platbu. Ke zrušení objednávky dojde v případě, že zákazník platbu nerealizoval vůbec či do dohodnutého termínu. Po zaplacení objednávky je uzavřena práce s ní a je převedena do stavu dokoncena.

10 Interpretace diagramů

Cíl cvičení

Z předložených diagramů interpretovat jejich význam a zhodnotit kvalitu modelu.

10.1 Teoretický základ

Při modelování systému je důležité u jednotlivých diagramů, které vytvořil jiný analytik umět porozumět smyslu předloženého modelu. Na základě znalostí UML je pak možné "číst"diagramy vytvořené jiným autorem.

Při předložení diagramu je vhodné si zodpovědět několik otázek.

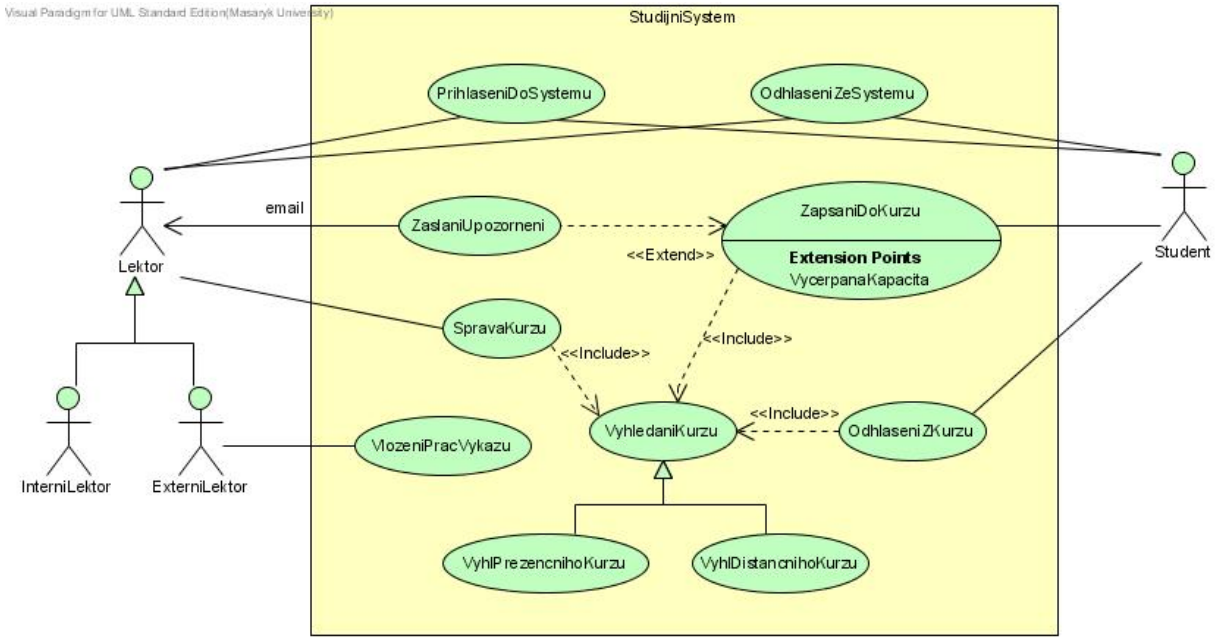
- Jsem schopen určit typ diagramu?
- Obsahuje diagram nějaké symboly, kterým nerozumím?
- Jsou v diagramu použity autorovy vlastní streotypy? Je patrný jejich význam?
- Rozumím všem použitým omezením v diagramu? Jsem schopen je převést do slovního popisu?

Na základě odpovědí mohu diagram buď okamžitě interpretovat nebo si musím dohledat případně neznámě symboly či si vyžádat další dokumentaci pro porozumění významu použitých prvků.

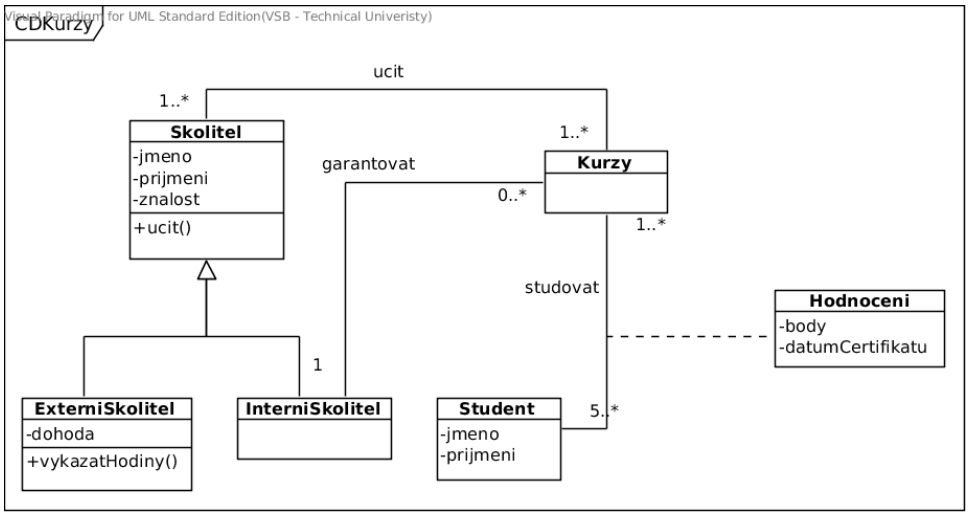
10.2 Úkoly

Pokuste se interpretovat uvedené diagramy. Nejprve na základě použitých symbolů určete, o jaký typ diagramu se jedná. Pak v diskuzi zjistěte, zda i ostatní studenti chápou význam diagramů obdobně, či zda je jejich interpretace odlišná a pokud ano, pak v čem.

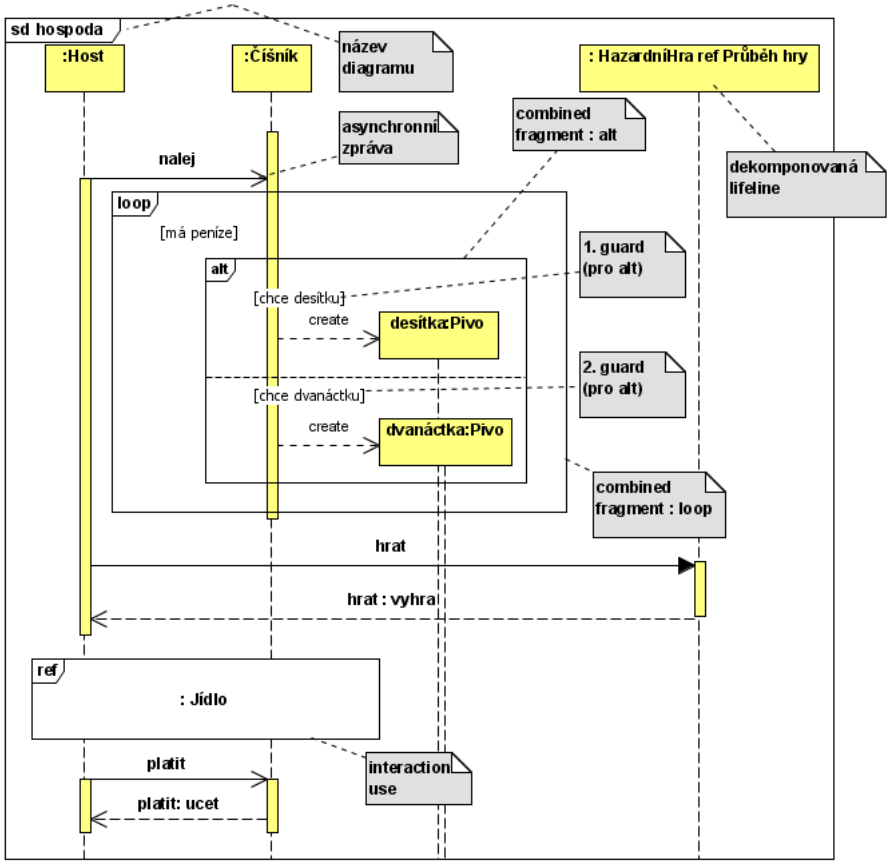
1. Interpretujte diagram na obrázku 22. Obrázek je převzat ze stránek `http://www.fi.muni.cz/~buhnova/PV167/03_Studium_UseCase.jpg`.
2. Interpretujte předložený diagram na obrázku 23.
3. Interpretujte diagram na obrázku 24. Obrázek je převzat ze stránek `http://mpavus.wz.cz/uml/img/seq-vp-3.png`.
4. Interpretujte diagram na obrázku 25. Obrázek je převzat ze stránek `http://uml.czweb.org/Diagramy/diagram_aktivit1.jpg`.



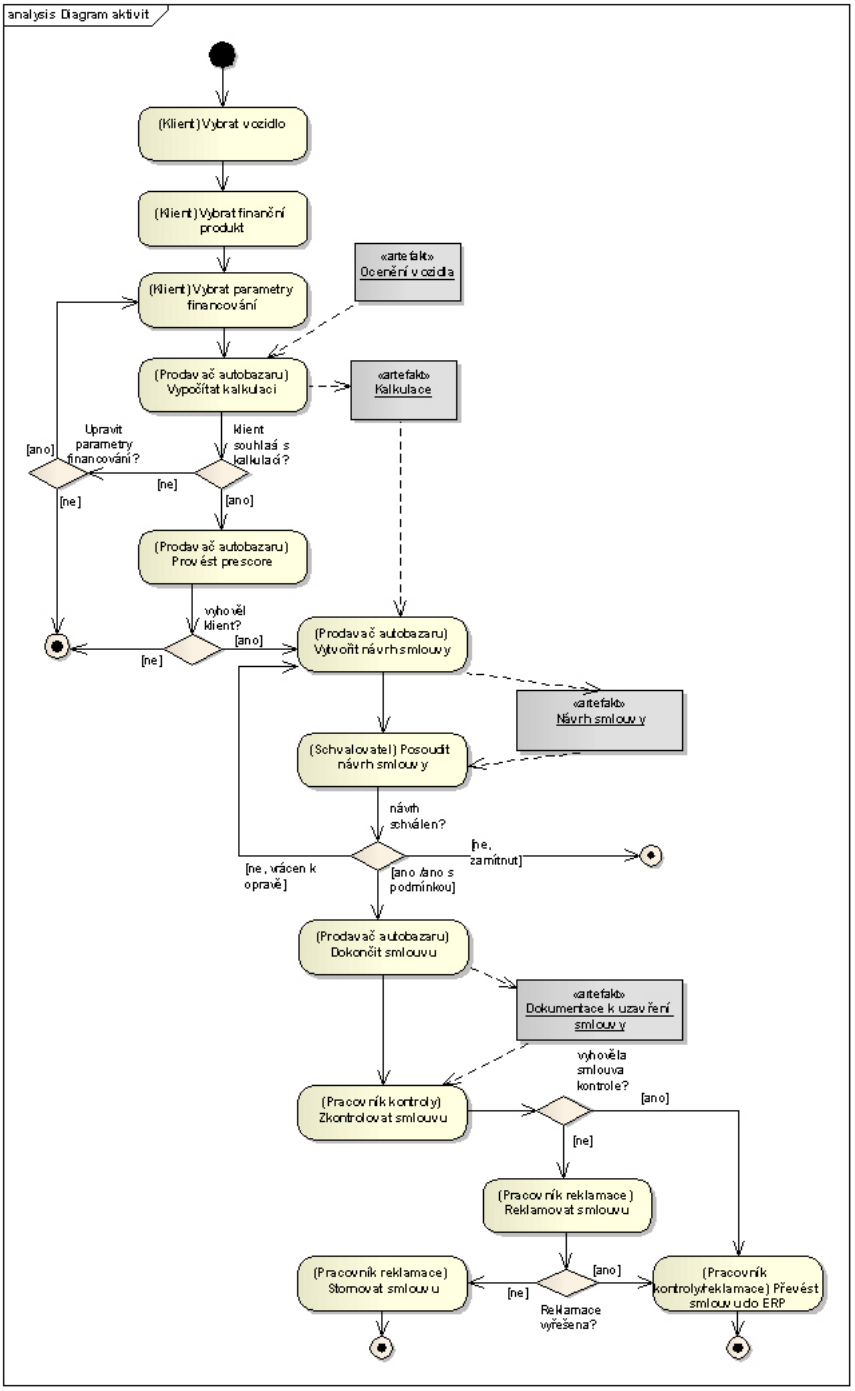
Obrázek 22: Diagram pro Studijní systém.



Obrázek 23: Stručný záznam systému pro evidenci kurzů.



Obrázek 24: Diagram systému Hospoda.



Obrázek 25: Diagram aplikace.

Reference

[1] Neustadt I. Arlow, J. *UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky.* Computer Press, 2007.

[2] M. Fowler. *UML Distilled: A Brief Guide to the Standard Object Modeling Language.* Addison-Wesley Professional, 2003.