## Analýza a návrh informačních systémů – zkouška, řádný termín, 4.1.2018

| 1 | 2 | 3a | 3b | 3c | 3d | 4 | 5 | Σ |
|---|---|---|---|---|---|---|---|---|
| /6 | /7 | /3 | /3 | /8 | /13 | /7 | /4 | /51 |

**Login:** **Jméno a příjmení:**

**Řada/pozice:** **Podpis:**

Odpovědi vypracovávejte přímo za text zadání. Hodnocena není délka odpovědi, nýbrž výstižnost ve vztahu k zadání. Pište dostatečně velkým písmem a čitelně. Dílčí body za odpovědi jsou orientační.

1. Popište, jakou roli hrají v architektuře orientované na služby (SOA) standardy. Uveďte, na kterých standardech jsou postaveny webové služby (Web-services) a co tyto jednotlivé standardy popisují a jaké konkrétně podporují základní vlastnosti/principy architektury SOA (které a jak). **6 bodů**

Standardy zajišťují kompatibilitu služeb nezávisle na způsobu jejich implementace. U služeb Web-service se jedná např. o XML a XML Schema Definition (XSD) pro o kompatibilitu reprezentace komunikovaných dat, Simple Object Access Protocol (SOAP) pro popis/protokol komunikace, Web Services Description Language (WSDL) pro popis rozhraní webových služeb, Universal Description, Discovery and Integration (UDDI) pro přístup k registrům webových služeb.

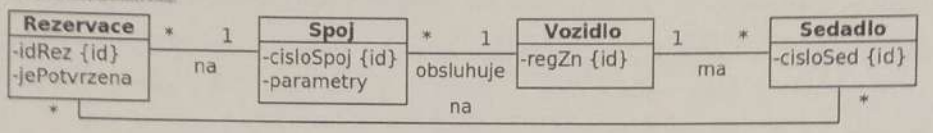
2. Uveďte alespoň tři principy agilního vývoje (např. z manifestu sdružení Agile Alliance) a vysvětlete je. Uveďte dvě charakteristiky modelu životního cyklu agilního vývoje, které tento model podle vás nejvýstižněji charakterizují, a vysvětlete je. Uveďte dvě techniky spojené s agilním vývojem (vlastního vývoje, ne řízení projektu) a uveďte jejich podstatu a proč jsou pro agilní vývoj důležité. **7 bodů**

Principy: uspokojení zákazníka pomocí CD; změny požadavků vítány; častý release; spolupráce vývojářů a zákazníků; motivovaní vývojáři s důvěrou a podporou; osobní setkávání; měřítko úspěchu je funkční SW; neustálý rovnoměrný vývoj; zaměření na dobré technické provedená a návrh; jednoduchost; samoorganizující se týmy; pravidelné vyhodnocení dosavadního způsobu práce a zlepšování

Charakteristiky: časté a krátké iterace pevné délky; všichni v týmu přispívají do všeho (vývojáři navrhují atp.)

Techniky: test-driven development (TDD); continous integration/delivery (CI/CD); agile modelling

3. Předpokládejte, že vyvíjíte informační systém pro prodej jízdenek. Pokud si chce zákazník koupit jízdenku, vyhledá pomocí systému vyhovující spoje, zarezervuje si jedno či více sedadel ve vozidle na vybraném spoji a po dokončení/potvrzení rezervace je tato zařazena do objednávky, kterou následně zaplatí. Vaším úkolem je realizovat základní scénář případu užití *rezervuj*, který rozšiřuje případ užití *hledejSpoje* o možnost zarezervovat si jedno či více sedadel ve vozidle na jednom z vyhledaných spojů. K dispozici máte již vytvořený dílčí model domény, který ukazuje koncepty a jejich vztahy, důležité pro vaše návrhová rozhodnutí. Atribut „parametry" třídy Spoj pro zjednodušení zapouzdřuje parametry spoje, podle kterých se vyhledává (výchozí a cílová stanice, časy aj.). Třída Vozidlo reprezentuje vozidlo obsluhující (nejen) daný spoj a mající sedadla (objekty třídy Sedadlo). Třída Rezervace pak představuje rezervaci sedadel v rámci daného spoje s atributem „jePotvrzena" indikujícím potvrzení dané rezervace.

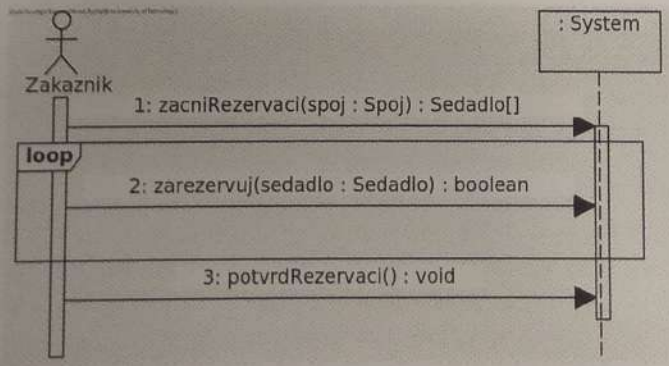



U scénáře *rezervujObjednej*, jehož realizaci navrhujete, je primárním aktérem zákazník a podmínkou je, že je přihlášený. Scénář má následující kroky:

1. Zákazník zahajuje výběrem funkce rezervace sedadel na zobrazeném spoji.
2. Systém zobrazí seznam sedadel ve vozidle na daném spoji a jejich stav (volné/obsazené).
3. Dále se do ukončení/potvrzení rezervace opakuje:
   3.1. Zákazník si vybere jedno ze sedadel a zvolí, že ho chce zarezervovat.
   3.2. Systém ověří, že je sedadlo aktuálně volné a pokud tomu tak je, sedadlo zarezervuje (přidá do rezervace), jinak požadavek odmítne.
4. Zákazník ukončí scénář potvrzením rezervace (resp. seznamu v ní rezervovaných sedadel).

Nezabývejte se tím, jestli by takový průběh rezervace byl uživatelsky přívětivý nebo ne, berte to jako požadavek daný tímto zadáním.

**3a)** Nakreslete **systémový diagram sekvence** scénáře, který má být realizovaný. **3 body**




**3b)** Popište **kontrakty systémových operací** pro kroky 1 až 3 výše popsaného scénáře ve tvaru:
*název_operace (parametry): předpoklad; stav_po_vykonání_operace* **3 body**

zacniRezervaci(spoj): byl vybrán konkrétní Spoj; existuje objekt Rezervace

zarezervuj(sedadlo): existuje objekt Rezervace a bylo zvoleno konkrétní Sedadlo; pokud bylo sedadlo v daném Spoji volné, je v Rezervaci

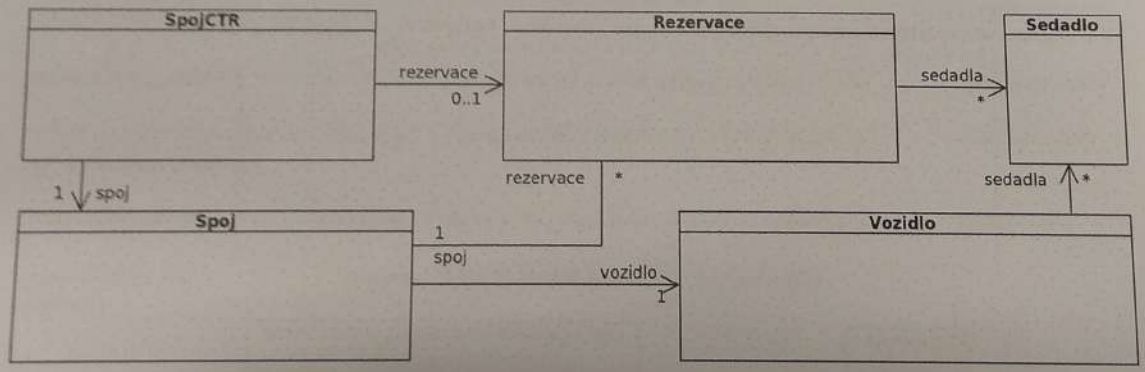
potvrdRezervaci(): existuje objekt Rezervace; hodnota atributu „jePotvrzena" u rezervace je pravda

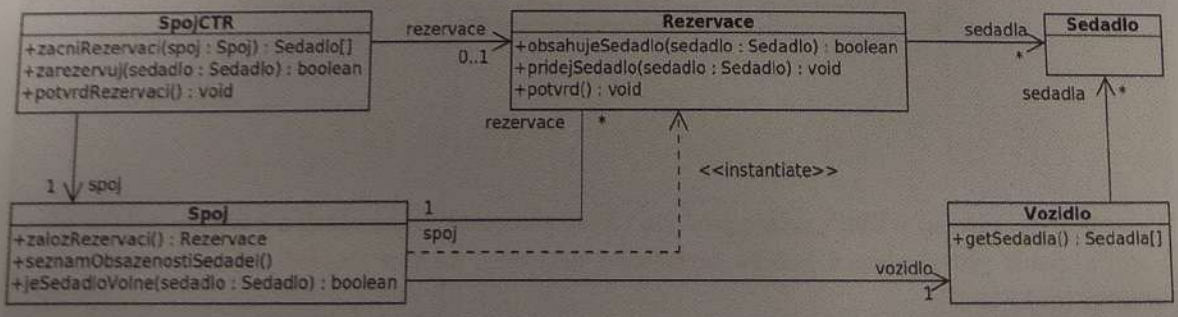
**3c)** Pro systémové operace z bodu 3b) **navrhněte realizaci** (řešíte pouze vrstvu domény, pro výstupy systémových operací jen dostupnost informace, nikoliv její zobrazení). Protože případ použití rozšiřuje jiný případ použití, jsou už navrženy některé třídy, jejich atributy dle modelu domény a operace (předpokládejte přinejmenším existenci getter a setter operací pro přístup k atributům) a jsou vytvořeny některé instance. Třídy můžete využívat, jsou-li vhodnými kandidáty na přiřazení zodpovědnosti, a přidávat u nich další potřebné atributy a operace. Jedná se o tyto třídy, případně instance (vizte též diagram v zadání k bodu 3d):

SpojCTR – řadič případu použití *hledejSpoje*, již vytvořena instance;
Spoj – reprezentuje zobrazený spoj, již vytvořena instance (vč. asociace na Vozidlo) a tato asociována s řadičem;
Vozidlo – reprezentuje vozidlo obsluhující (nejen) zobrazený spoj, již vytvořena instance (vč. kolekce sedadel);
Sedadlo – reprezentuje sedadlo v daném vozidle, již vytvořena instance;
Rezervace – reprezentuje rezervaci daných sedadel na daný spoj, dosud není vytvořena instance;

Nejprve strukturovaným způsobem podobným CRC štítkům rozhodněte o **přiřazení zodpovědnosti třídám** (a jim odpovídajícím operacím) a o **spolupracujících třídách**. Každé přiřazení zodpovědnosti **zdůvodněte** (PROČ jste přiřadili danou zodpovědnost právě této třídě, ne co operace dělá). Pokud použijete některý princip GRASP, nestačí uvést jenom jeho název, ale i tady musí být zřejmé, proč. Není zde nutné uvádět detailně zodpovědnosti za vytvoření kolekce, zařazení do kolekce, čtení z kolekce, čtení a nastavování hodnot atributů (stačí až v diagramu v 3d) a není třeba uvádět typy parametrů operací, stačí vhodná jména. **Použijte pro tyto účely tabulku na samostatném listu na str. 5.** **8 bodů**

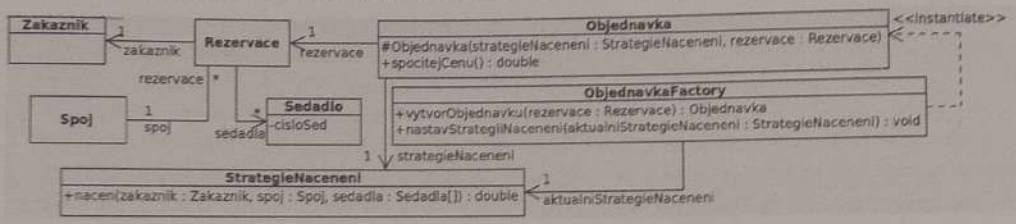
**3d)** Doplňte **návrhový diagram tříd**, obsahující již navržené třídy, o část, kterou jste navrhli na základě tabulky z 3c). Musí obsahovat všechny navržené operace (parametry a návratové hodnoty není třeba uvádět) a atributy (za 4 body). Dále nakreslete **diagramy sekvence nebo komunikace** pro navržené systémové operace a zajištění dostupnosti jejich návratových hodnot (za 9 bodů). **Diagramy sekvence nebo komunikace nakreslete na samostatném listu na str. 6.** **13 bodů**

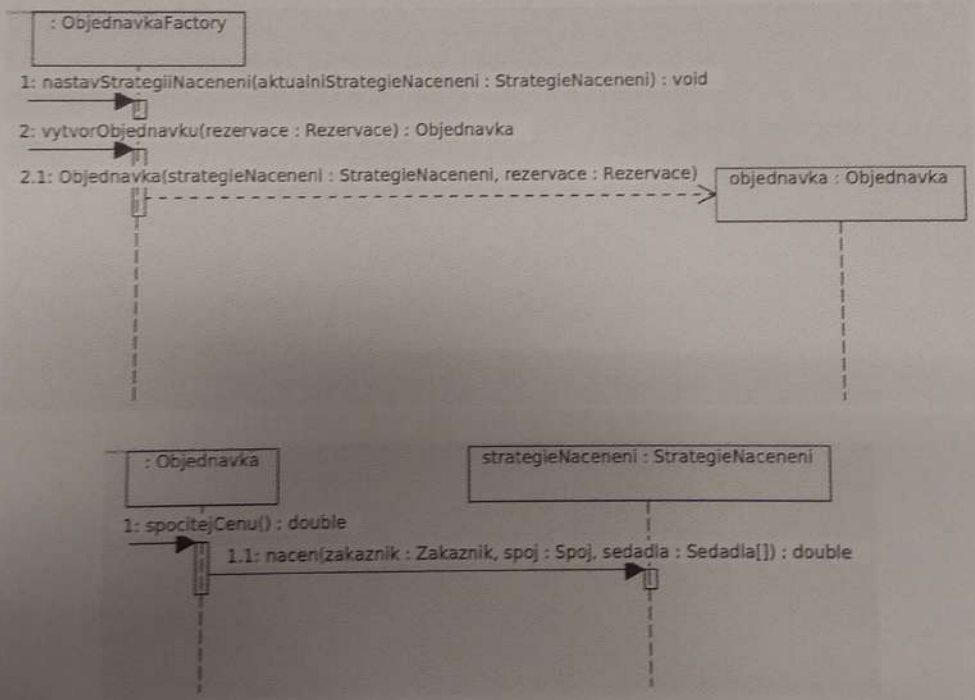


4. Předpokládejte, že pokračujete ve vývoji informačního systému pro prodej jízdenek z bodu 3). Vaším úkolem je **navrhnout realizaci výpočtu ceny objednávky** vycházející z potvrzené rezervace (pro daný spoj a dané sedadla) a vlastností objednávajícího zákazníka. Při analýze jste zjistili, že výpočet ceny ovlivňuje řada aspektů: např. druh sedadel (výbava), počet sedadel (množstevní sleva), vytíženost spoje (poslední volná sedadla jsou dražší), kategorie zákazníka (slevová karta) a další. Mimo to se způsob výpočtu často mění dle potřeb prodejce (např. sezónní akce). Z dosavadního návrhu realizace můžete předpokládat, že celková cena je počítána operací *spocitejCenu()* třídy *Objednavka*, která je asociovaná s třídami *Rezervace* a *Zakaznik* (tedy objekt objednávky zná objekt potvrzené rezervace a objekt zákazníka, ať už přímo či přes rezervaci).

Navrhněte obecnou realizaci operace *spocitejCenu()*, kde použijete vhodný návrhový vzor související s touto operací (operaci můžete doplnit parametry). Uveďte název vzoru (za 1 bod) a jeho použití v této konkrétní situaci namodelujte v návrhovém diagramu tříd a objasněte formou diagramu sekvence (za 3+3 body). **7 bodů**




5. Vysvětlete, co rozumíme v UML „stereotypem" (za 1 bod), jakou má v jazyce notaci a k čemu slouží (za 1,5 bodů) a uveďte příklad alespoň jednoho konkrétního stereotypu, který je součástí jazyka, a uveďte, co označuje (za 1,5 bodů). **4 body**

**Login:** **Jméno a příjmení:**

**Řada/pozice:** **Podpis:**

**Strana 5** (pro přiřazení zodpovědnosti třídám a spolupracujících třídách z bodu 3c)

| Popis zodpovědnosti, Operace | Třída | Zdůvodnění přiřazení zodpovědnosti dané třídě | Spolupracující třídy |
|---|---|---|---|
| **zacniRezervaci(spoj)** | SpojCTR | Již vybrána jako řadič případu použití (CTR) | Spoj |
| *vytvoření instance třídy Rezervace, její inicializace* zalozRezervaci() | Spoj | Bude používat vytvořené rezervace pro kontroly obsazenosti sedadel, zná inicializační parametr rezervace (sebe sama; CRE) | Rezervace |
| *získání seznamu sedadel s inforamcí o obsazenosti* seznamObsazenostiSedadel() | Spoj | Zná jako vlastní atribut/asociace vozidlo obsluhující daný spoj a mající sedadla a umí zjistit, je-li sedadlo volné (IE) | Vozidlo, Spoj |
| *získání seznamu sedadel* getSedadla() | Vozidlo | Vlastní kolekci sedadel ve svém atributu (IE) | List<Sedadlo> |
| *získání informace je-li sedadlo někde volné* jeSedadloVolne(sedadlo) | Spoj | Zná jako vlastní atribut/asociace již existující rezervace k danému spoji (IE) | List<Rezervace> Rezervace |
| *získání informace je-li sedadlo rezervováno* obsahujeSedadlo(sedadlo) | Rezervace | Zná jako vlastní atribut/asociace v rezervaci zarezervovaná sedadla (IE) | List<Sedadlo> |
| **zarezervuj(sedadlo)** | SpojCTR | Již vybrána jako řadič případu použití (CTR) | Spoj, Rezervace |
| *přidá sedadlo do rezervace* pridejSedadlo(sedadlo) | Rezervace | Vlastní kolekci zarezervovaných sedadel (IE) | List<Sedadlo> |
| **potvrdRezervaci()** | SpojCTR | Již vybrána jako řadič případu použití (CTR) | Rezervace |
| *potvrdí rezervaci* potvrd() | Rezervace | Vlastní atribut indikující potvrzení (IE) | |

**Strana 6** (pro diagramy sekvence nebo komunikace z bodu 3d)