

**SE201 UVOD U SOFTVERSKO INŽENJERSTVO**

**Januar 2020**

**Pravila polaganja:**

U školskoj 2019/2020. godini završni ispit za predmet SE201 polaže se na sledeći način:

Ispit iz predmeta SE201 Uvod u Softversko inženjerstvo održava se u učionicama UM-a. Na računarima u učionici biće instalirano NetBeans razvojno okruženje kao i Power Designer alat. **Na ispitu nije dozvoljeno korišćenje literature, kao ni LAMS-a. Na ispitu nije dozvoljeno korišćenje komunikacionih tehnologija: mobilni telefon, chat program (gTalk, Skype, Facebook, Google+ i drugi), email kao i posedovanje i korišćenje USB diska.** Odnosno, nije dozvoljena ni jedna vrsta komunikacije ni sa ostalim studentima u učionici niti sa bilo kim van učionice tokom trajanja ispita. **Trajanje ispita je 3 sata.**

**Ispit se sastoji iz dva dela.** U prvom delu student odgovara na 4 pitanja, bez uključenih računara, a odgovore na pitanja student piše na papiru. U drugom delu ispita student radi ispitni zadatak. **Pitanja donose maksimalno do 10 poena, a zadatak do 20 poena.** **Student je položio ispit ako položi oba dela, tj. da ima najmanje 5 poena na teorijskom delu, i najmanje 10 poena na praktičnom delu ispita**

Student može da počne da radi ispitni zadatak tek kada preda svoje odgovore na pitanja dežurnom asistentu. Power Designer modele, dokument sa odgovorima na dobijen ispitni zadatak treba poslati na e-mail asistenta.

**Pitanja:**

Izberite 4 od 5 datih pitanja i dajte odgove.

1. Šta je inkrementalni razvoj softvera? Zašto se koristi? Šta je inkrement? Koje su prednosti inkrementalnog razvoja u odnosu na model vodopada? Kako se primenjuje inkrementalni razvoj softvera? Koji su problemi u primeni?
2. Šta su nefunkcionalni zahtevi? Zašto su oni važni? Navedite osnovne grupe nefunkcionalnih zahteva i opišite svaki. Koji se problemi javljaju kod utvrđivanja nefunkcionalnih zahteva? Koja je metrika nefunkcionalnih zahteva? Koje su teškoće u njenoj primeni?
3. Šta klasa tipa Control? Kako se utvrđuje klasa tipa Control analizom slučajeva korišćenja? Navedite primer.
4. Šta je šablon? Šta je šablon projektovanja? Koje informacije sadrži opis jednog šablona za projektovanje. Opišite šablon „posmatrač“ (engl. Observer). Kada se koristi ovaj šablon? Dajte primer primene ovog šablona.
5. Šta je evolucija softvera? Kako izgleda spiralni model razvoja i evolucije softvera? Ko održava softver razvijen po narudžbini?

**Zadaci:**

Potrebno je razviti aplikaciju za deljenje vožnje. Korisnici koji koriste aplikaciju moraju prvo da se registruju, a zatim i prijave. Svaki korisnik ima opciju da pronađe vožnju ili da ponudi prevoz. Kada traži vožnju korisnik unosi grad iz kojeg polazi i grad u koji ide, kao i datum. Prikazuju mu se vožnje za za tu relaciju i taj datum. Za svaku vožnju se ispisuje vreme i mesto polaska, kao i okvirno vreme dolaska. Korisnik može mesto polaska i dolaska da vidi u okviru mape. Pored toga, korisnik vidi koliko ima slobodnih mesta u vozilu, kao i ocenu vozaca. Korisnik moze da zahteva da se priključi voznji. Priključivanje vožnji mora da odobri vozac.

Na kraju svake vožnje, putnici mogu da ocene vozača, tj. kako su zadovoljni vožnjom. Isto tako i vozač moze da oceni putnike, tj. njihovo ponašanje tokom vožnje.

Korisnik koji nudi vožnju mora da unese podatke o automobilu (model, proizvođač, boja, broj sedišta, postavi sliku sebe i sliku automobila). Pored toga, vozač može da stavi i neke dodatne informacije, na primer: zabranjeno pušenje, dozvoljena muzika, pričanje sa vozačem, kućni ljubimci itd.

Sve vožnje se prate pomoću GPS sistema. U svakom trenutku korisnik aplikacije može da zahteva pomoć, kao i da prijavi ugrožavanje bezbednosti. Zaposleni na kontroli i održavanju sistema mogu da blokiraju naloge vozača koji imaju veoma niske ocene ili koji ugroze bezbednost putnika.

Za ovaj problem, treba da uradite sledeće zadatke:

1. **(3 poena)** Definišite model zahteva. Navesti sve funkcionalne i nefunkcionalne zahteve i ukratko ih opisati.
2. **(2 poena)** Definišite arhitekturu sistema i kratko navedite funkciju svake komponente te arhitekture. Obrazložite zašto ste izabrali baš takvu arhitekturu.
3. **(3 poena)** Definišite UML slučajeve korišćenja celokupnog sistema
4. **(3 poena)** Napišite primarne i sekundarne scenarije (opis, aktor, preduslovi, normalni tok događaja, izuzeci i postuslovi) **za priključivanje korisnika postojećoj vožnji** i nacrtajte prateće sistem sekvencijalne dijagrame (ovi sekvencijalni dijagrami su na najvećem nivou apstrakcije, jer pokazuju samo intetrakciju aktera i prikazanih slučajeva korišćenja).
5. **(4 poena)** Izvršite detaljno projektovanje sistema, tj. nacrtajte sekvencijalne dijagrame (primarni i sekunadarni scenario) najveće detaljnosti (nivo upotrebljenih klasa) za slučaj korišćenja iz tačke 4. Za svaku poruku definišite argumente (tip i naziv podatka), a poruku treba prikazati kao metod u klasnom dijagramu (sa svojim nazivom, povratnom vrednošću i svojim argumentima).
6. **(3 poena)** Prikažite UML dijagram svih klasa koje učestvuju u konačnim sekvencijalnim dijagramima iz tačke 5. Svaka klasa mora da sadrži sve atribute i metode definisane primenom sekvencijalnih dijagrama iz tačke 5. Za svaki metod i atribut navedite i pravo pristupa. Ne morate navoditi i prikazivati geter i seter metode.
7. **(2 poena)** Primeniti ***Visitor*** pattern gde treba proći kroz sve vožnje u poslednjih mesec dana i izračunati najveću, najmanju i prosečnu ocenu vozača.

Pri radu, koristite Power Designer.