Datum: 13.03.2018.

1. Analiza i dizajn

Analiza i dizajn su uvijek prvi koraci u razvojnom procesu softvera. Cilj **analize** je istraživanje problema i zahtjeva sistema (a ne rješenja), dok **dizajn** ima za cilj identifikaciju konceptualnog rješenja problema koji će zadovoljiti postavljene zahtjeve. Na primjer, rezultat dizajna je šema baze podataka i softverski objekti (dizajn objekata ili dizajn baze podataka). Naposljetku, rezultat faze dizajna se implementira.

Pri **objektno orijentisanoj analizi** naglasak je na pronalasku i opisu objekata (ili koncepata) iz domene problema i veza među njima, dok **objektno orijentisani dizajn** stavlja naglasak na pronalazak softverskih objekata i načina na koji oni komuniciraju u cilju zadovoljavanja zahtjeva.

Poznavanje objektno orijentisanog programskog jezika i UML notacije nije dovoljno za razvoj kvalitetnog softvera. Potrebno je ovladati ***vještinom*** kreiranja dobrog objektno orijentisanog dizajna.

Koje klase će postojati u sistemu? Koja zaduženja će imati pojedina klasa? Na koji način će objekti komunicirati? Sve su to ključna pitanja na koja treba dati odgovore prilikom objektno orijentisane analize i dizajna.

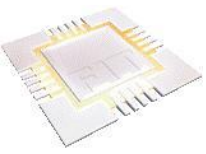
Objektno orijentisanom dizajnu prethodi analiza zahtjeva koja najčešće uključuje pisanje slučajeva upotrebe ("use-case") i crtanje "use-case" dijagrama.

2. Use case model

Use case model (model slučajeva upotrebe) je rezultat definisanja informacijskih zahtjeva sistema (OOA) i sastoji se od skupa use case dijagrama (slučajeva upotrebe ili korištenja). Skup slučajeva upotrebe predstavlja sve pretpostavljene načine korištenja sistema, tj. određuje ponašanje sistema ili njegovog dijela i opisuje funkcionalnosti sistema koje zahtijevaju korisnici. Slučajevi upotrebe predstavljaju odgovor na pitanje što je to što sistem treba da radi. Znači, slučajevi upotrebe opisuju jedan od načina korištenja sistema i interakciju između korisnika i sistema.

Use case nije objektno orijentisani artefakt, već predstavlja alat za definisanje zahtjeva, kojim se predstavlja funkcionalnost sistema. *Use cases* opisuju ponašanje sistema sa korisničkog aspekta, po principu akcije i reakcije (Ivar Jacobson). Naglasak je na tome *šta* sistem radi, a manje na koji način to radi. Ovaj artefakt UML-a nam omogućuje odrediti okvire sistema i definirati odnose između sistema i okruženja u kojem se nalazi. Slučajevi upotrebe se koriste da bi se ostvarilo željeno ponašanje sistema koji se razvija, bez obaveze da se odrede načini realizacije tog ponašanja. Budući korisnici sistema bi ih trebali razumjeti i ocijeniti.

Svaki ***"use-case"*** se sastoji od ***dijagrama*** i ***dokumenta koji ga opisuje***. Najvažniji dio slučaja upotrebe jeste ***dokument koji ga opisuje***.



"Use-case" se imenuje slično korisničkom cilju (npr. za funkcionalnost prodaje knjiga imali bi "use-case" "Prodaja knjiga"). Naziv slučaja upotrebe (use-case) uvijek počinje glagolom. Postoje slučajevi kada se više ciljeva može predstaviti jednim "use-case"-om. To je obično slučaj kada imamo ciljeve tipa CRUD ("create", "retrieve", "update", "delete") za neki element (npr. dodavanje korisnika, pronalazak korisnika, izmjena korisnika, brisanje korisnika). Sva četiri cilja možemo predstaviti jednim "use-case"-om koji obično imenujemo u obliku "Upravljanje <X>" (npr. "Upravljanje korisnicima").

Dijagrami slučajeva upotrebe (use case) se definiraju na osnovu *funkcionalnog modela sistema* i opisuju interakciju objekata koji se nalaze van sistema sa samim sistemom. Dijagram slučajeva upotrebe je graf čiji su čvorovi učesnici i slučajevi upotrebe međusobno povezani.

Scenario je primjer izvedbe slučaja upotrebe (instanca slučaja upotrebe).

Razvoj dijagrama slučajeva upotrebe definira se sljedećim aktivnostima:

- definiranjem učesnika,
- definiranjem slučajeva upotrebe,
- definiranjem tipova veza između učesnika i slučajeva upotrebe i
- izradom dijagrama slučaja upotrebe.

2.1. DEFINIRANJE UČESNIKA

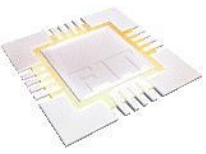
*Učesnik (aktor) j*e objekt koji se nalazi van sistema i predstavlja jedan tip korisnika.

Učesnici su bilo ko ili bilo što, što će komunicirati sa sistemom koji se razvija. Učesnik je tip, a ne instanca individualnog korisnika i ni u kom slučaju ne odgovara individualnom korisniku sistema. Učesnik može da bude korisnik (čovjek), neki drugi sistem ili čak vrijeme (funkcija koja se aktivira u određeno vrijeme).

Korisnik je čovjek koji koristi sistem, dok je učesnik specifična uloga koju korisnik ima u komunikaciji sa sistemom. Jedan korisnik može predstavljati više učesnika, a više učesnika jednog korisnika i jedan učesnik više korisnika. Učesnik komunicira sa sistemom putem poruka.

Učesnik je jedna vrsta klase (tip) i može da budu u svim relacijama koje važe za klase. Učesnike je moguće identificirati na osnovu odgovora na sljedeća pitanja:

- Ko će koristiti osnovnu funkcionalnost sistema (primarni akteri)?
- Kome će biti potrebna podrška sistema u obavljanju dnevnih zadataka?
- Ko treba da upravlja, administrira i održava sistem (sekundarni akteri)?
- Kojim hardverskim uređajima treba da upravlja sistem?
- S kojim drugim sistemima dotični sistem treba da bude u vezi? Ti sistemi mogu da se podijele na sisteme koji će inicirati komunikaciju sa našim sistemom i na one koje će naš sistem kontaktirati. Sistemi mogu biti računarski sistemi, kao i druge aplikacije na računaru u kojima se sistem nalazi.
- Ko ili zašto je zainteresovan za rezultate koje sistem proizvodi?



Učesnici se u dijagramu predstavljaju na sljedeći način:

Slika 1: Predstavljanje učesnika u use-case dijagramima

Postoje tri tipa učesnika:

- **primarni učesnici** koristite osnovne funkcionalnosti sistema u direktnoj interakciji sa sistemon,
- **sekundarni učesnici** administriraju i održavaju sistem i
- **učesnici inicijatori** nisu u direktnoj interakciji sa sistemom, ali na njihovu inicijativu primarni akteri iskorištavaju funkcionalnosti sistema (npr. kupci na kasi).

Primarni učesnik slučaja upotrebe jeste *stakeholder* (zainteresirana strana)[1] koji poziva sistem za isporuku jednog od njegovih servisa ili usluga. To je često, ali ne uvijek, učesnik koji aktivira use case.

Sekundarni učesnik je neko čija je pomoć neophodna sistemu da bi se postigao cilj primarnog aktera.

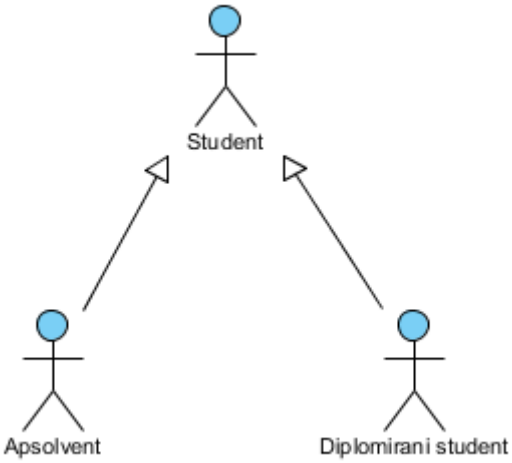
Primjer

- Use case: Obrada kreditnog zahtjeva
- Primarni učesnik: Kreditni službenik
- Sekundarni učesnik: Sistem za ocjenu kreditne sposobnosti

Važna relacija je *generalizacija*, gdje izvedeni učesnik ima sve osobine osnovnog učesnika i može da učestvuje u svim slučajevima upotrebe kao i osnovni učesnik.

UML također klasificira učesnike na „*konkretne*" i „*apstraktne*". Ulogu apstraktnih aktera čini skup zajedničkih odgovornosti konkretnih aktera. Ova dva tipa aktera se povezuju vezama generalizacije.
Npr. Ljekar (kardiolog, zubar, internista - pregleda pacijente)

[1] Osoba, grupa ili organizacija koja ima interes u vašoj organizaciji.



Slika 2: Primjer veze generalizacije

2.2. DEFINIRANJE SLUČAJEVA UPOTREBE

Definiranje slučajeva upotrebe opisuje karakteristične sekvence akcija u tipičnim situacijama upotrebe sistema, što znači da je to tehnika kojom se predstavlja poslovni proces s korisničkog aspekta ili scenarij koji opisuje skup sekvenci događaja. Slučajevi upotrebe grafički se predstavljaju *elipsom* i *nazivom* slučaja upotrebe.

Naziv slučaja upotrebe treba da bude jasan i ne dugačak (obično se upotrebljava glagolski oblik).

Slučaj upotrebe je potrebno tekstualno opisati. U opisu se detaljno specificira *šta* taj slučaj upotrebe *radi*. Mogući template use case dokumenta (opisuje use case) dat je sljedećom tabelom:

Use case:	Naziv slučaja upotrebe	**ID:**	UC1
Autor:	eMIN@		
Datum:			
Akteri:	Akteri s kojima je u vezi		
Opis (description):	Kratak opis slučaja upotrebe		
Preduslovi (preconditions):	Uslovi koji moraju biti ispunjeni prije instanciranja slučaja upotrebe		
Rezultati (postconditions):	Opis posljedica izvršenja slučaja upotrebe		
Osnovni tok događaja (main flow):	Najčešći tok interakcije sa sistemom		
Alternativni tok(ovi) događaja:	Opis mogućeg alternativnog toka interakcije sa sistemom		
Exeption(s) flow:	Opis mogućeg nepredviđenog toka događaja.		

2.3. DEFINIRANJE TIPOVA VEZA IZMEĐU UČESNIKA I SLUČAJEVA UPOTREBE

U okviru UML standarda definiraju se sljedeće veze:

- asocijacija (association)
- asocijacija između slučajeva upotrebe
  <<include>> ili <<extend>>
- generalizacija (generalization-inheritance)

Slučaj upotrebe može biti povezan sa učesnikom ili sa drugim slučajevima upotrebe. Tip veze između učesnika i slučaja upotrebe je asocijacija. ***Asocijacija*** je dvosmjerna veza koja spaja učesnike i slučajeve upotrebe i predstavlja komunikaciju između njih. Asocijacija između samih učesnika, ili slučajeva upotrebe, definira logičku povezanost tih elemenata.

Veza između dva slučaja upotrebe predstavlja njihovu zajedničku funkcionalnost. Zajednička funkcionalnost je predstavljena jednim slučajem upotrebe, koji je povezan sa slučajevima upotrebe koji je koriste. Tip veze između slučajeva upotrebe može biti: uključenje <<include>> i proširenje <<extend>>. Vezom uključenje se povezuju dva slučaja upotrebe na način da jedan slučaj upotrebe u toku svog izvođenja u potpunosti izvede uključeni slučaj upotrebe. Vezom proširenje se povezuju dva slučaja upotrebe, pri čemu jedan proširuje funkcionalnost drugog. Proširenje se ostvaruje ako je zadovoljen određeni uslov u tački proširenja.

2.4. KREIRANJE VEZA ASOCIJACIJE IZMEĐU OBJEKATA

Veze asocijacije se kreiraju između slučajeva korištenja i učesnika. Mogu biti nacrtane u dva smjera, ovisno o tome da li je akter kojeg povezujemo primarni ili sekundarni:

- ukoliko se radi o primarnom učesniku veza asocijacije je usmjerena od aktera ka slučaju upotrebe, a
- ukoliko je to sekundarni učesnik veza asocijacije je usmjerena od slučaja upotrebe ka akteru.

2.5. ZADACI

Riješeni primjeri će primarno biti bazirani na jednoj tematici tokom cijelog semestra, tačnije softverskom projektu ***eUniverzitet***. Kratak opis njegovih osnovnih elemenata je dat u nastavku zajedno sa prvim primjerom izrade use-case dijagrama za predstavljeni problem.

eUniverzitet

eUniverzitet platforma primarno je namijenjena studentima u cilju pružanja blagovremenih informacija o studiju i pristupa svim relevantnim nastavnim sadržajima. Pored studenata, korisnici ove softverske platforme su i uposlenici univerziteta, odnosno nastavno osoblje i referenti. Riječ je zapravo o sistemu za podršku učenju ili, kako se on u literaturi često naziva, LMS-u (*eng. Learning Management System*). Osnovni moduli ovog softvera su ukratko opisani u nastavku:

- **eSluzba** – Modul namijenjen studentskoj službi fakulteta čiji je primarni fokus matična knjiga studenta sa mogućnošću upisa, evidencije uplata, upisa i ovjere semestara, administracije predmeta i ovjere ocjena, te izdavanja uvjerenja.
- **eNastava** – Modul za nastavno osoblje (nastavnici, asistenti, demonstratori) koji se zasniva na objavi nastavnih sadržaja i obavještenja studentima, evidenciji ispitnih obaveza i zaključivanju ocjena.
- **eStudent** – Studentski modul koji u osnovi pruža potpuni elektronski uvid u informacije o studiju (upisi, uplate, nastavni plan, uspjeh i td.), kao i mogućnost preuzimanja nastavnih materijala i pregleda obavještenja, odnosno detaljnog praćenja pojedinih predmeta nastavnog plana.

Za ispravnost pojedinih modula opisane softverske platforme zadužen je administrator sistema.

Zadatak 1: Funkcionalni zahtjevi – eSluzba

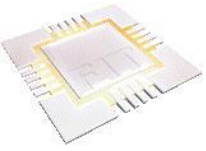
Osnovni korisnici modula eSluzba su referenti studentske službe, koji se za pristup sistemu prvobitno moraju prijaviti sa dodijeljenim korisničkim imenom i lozinkom. Poslovanje studentske službe jeste primarno administracija matične knjige studenata. Međutim, prije detaljnog opisa mogućnosti unutar elektronske matične knjige, potrebno je da budu zadovoljeni određeni preduslovi koji također spadaju u opis posla referenta studentske službe.

Kako bi upis studenata mogao biti u potpunosti podržan, neophodno je da referent evidentira sve predmete nastavnog plana i programa pojedinih odsjeka fakulteta, te da obavi detaljnu evidenciju nastavnog plana koja uključuje unos sljedećih podataka: semestar, naziv, tip i šifru predmeta, te ECTS bodove. Za svaki predmet nastavnog plana se opcionalno može evidentirati uslovljenost, odnosno odabrati lista uslovljenih predmeta. Također, svaki nastavni plan definiše tzv. matricu kompetencija gdje se svakom predmetu pridružuje jedna ili više kompetencija koju studenti stiču po završetku studija.

Po završetku unosa podataka o nastavnom planu, referent može da obavlja upis studenata kroz unos ličnih i kontakt podataka, podataka o prethodnom obrazovanju studenta, statusnih podataka upisanog studija, te opcionalno i podataka o uplatama. Sistem po završetku upisa studentu automatski dodjeljuje predmete odabranog nastavnog plana na upisanoj godini studija, te vrši automatski upis semestra i dodjelu pristupnih podataka. Administracija matične knjige se naknadno vrši putem pretrage po većem broju kriterija (odsjek, smjer, nastavni plan, broj indeksa, ime i prezime studenta), uz mogućnost pregleda i izmjene podataka o studentu, upisa i ovjere semestara, upisa i ovjere ocjena, te evidencije uplata. Svakim upisom ocjene, sistem studentu šalje obavještenje putem e-mail-a.

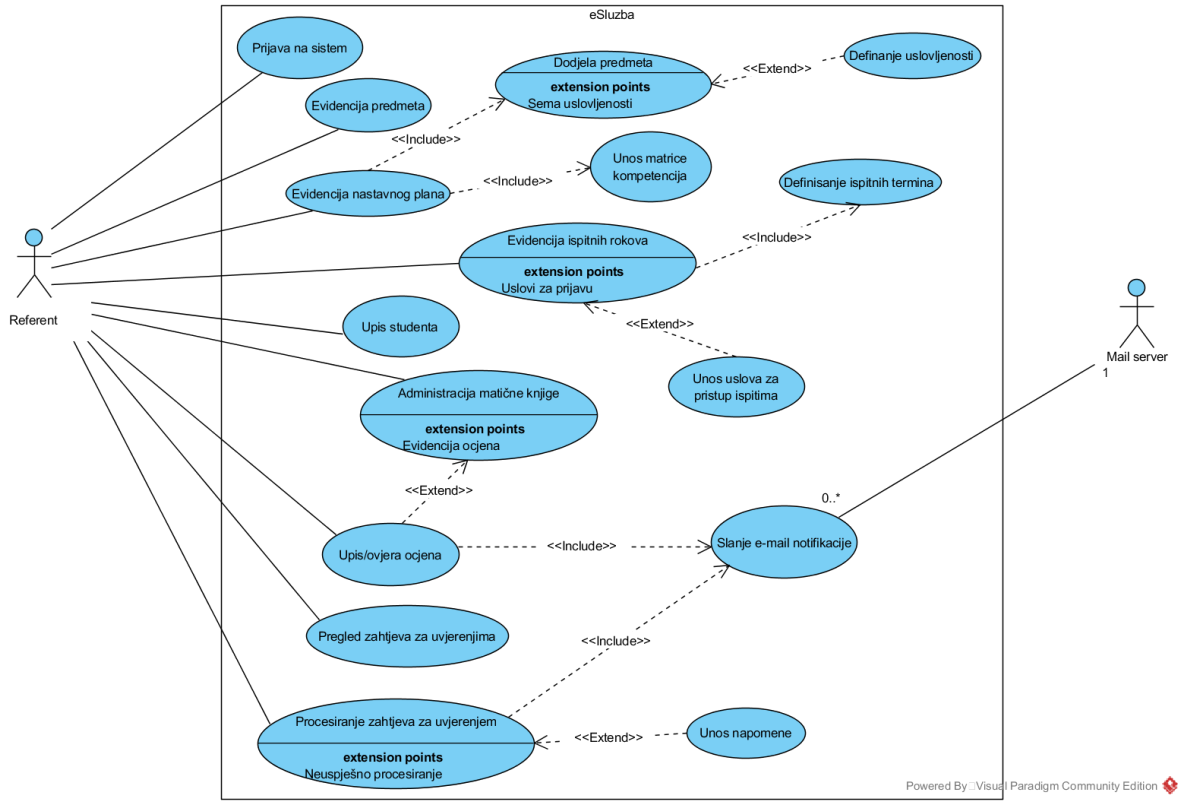
Osim navedenih, važan zadatak referenta za studentska pitanja jeste i izdavanje uvjerenja, gdje je moguće pregledati sve aktivne zahtjeve studenata, te uspješno ili neuspješno procesirati te zahtjeve. Uspješno procesiranje podrazumijeva printanje izvještaja i slanje notifikacije studentu putem softvera i na e-mail, dok neuspješno procesiranje podrazumijeva unos razloga zbog kojeg uvjerenje nije moguće izdati i također slanje notifikacije studentu.

Pored navedenih, zadaci studentske službe uključuju i evidenciju ispitnih rokova i ispitnih termina, gdje se za svaki ispitni rok mogu po potrebi definisati i dodatni uslovi koji određuju mogućnost elektronske prijave ispita (npr. minimalni uplaćeni iznos, ovjeren semestar, godina studija, broj položenih ispita i td.).



a) Kreirati odgovarajući use-case dijagram za prikaz modula studentske službe.

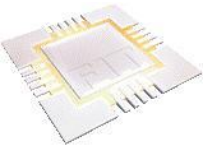
Prijedlog rješenja:



a) Uraditi detaljnu specifikaciju slučaja korištenja „Upis studenta".

Use-case	**Upis studenta**	**ID:**	UC07
Autor:	Larisa		
Datum:	08.02.2017.		
Akteri:	Referent		
Opis (description):	Referent studentske službe unosi podatke o upisanom studentu koji su grupisani na sljedeći način: lični podaci, kontakt podaci, podaci o prethodnom obrazovanju, statusni ili podaci o studiju, te opcionalno i podaci o uplatama. Po obavljenom upisu, student dobiva mogućnost pristupa softverskoj platformi.		
Preduslovi (preconditions):	Referent prijavljen na sistem. Uneseni podaci o nastavnom planu.		
Rezultati (postconditions):	Podaci o novom studentu su pohranjeni u bazu podataka. Studentu je omogućen pristup predmetima nastavnog plana na upisanoj godini studija.		
Osnovni tok događaja	1. Referent bira opciju za upis novog studenta. 2. Sistem omogućava unos podataka kroz tri koraka: a) Unos ličnih, kontakt i podataka o obrazovanju studenta b) Unos statusnih ili podataka o studiju c) Unos podataka o uplatama 3. a) Referent unosi lične podatke o studentu: • Ime (Ime roditelja) i Prezime • JMBG i Datum rođenja • Spol • Država, Opština i Mjesto rođenja		

	• Državljanstvo i Nacionalnost Referent unosi kontakt podatke o studentu: • Država, Opština, Mjesto i Adresa boravka • Mjesto i Adresa prebivališta • Kontakt telefon • E-mail adresa Referent unosi podatke o obrazovanju studenta: • Škola • Stepen obrazovanja i Zvanje • Datum i Ak. godina maturiranja • Status zaposlenja • Napomena Referent potvrđuje unos podataka sa prvog koraka i prelazi na naredni korak. Referent unosi statusne podatke studenta: • Fakultet/Studij • Studijski program / Odsjek • Smjer • Broj indeksa • Nastavni plan i Upisani semestar • Način studiranja i Status • Datum i Ak. godina upisa Referent potvrđuje unos podataka sa drugog koraka i prelazi na naredni korak ili potvrđuje upis studenta. Referent (opcionalno) unosi podatke o uplatama studenta: • Datum i Ak. godina • Svrha i Broj uplate • Godina studija i Semestar • Iznos • Napomena Referent potvrđuje upis studenta. 4. Sistem pohranjuje podatke o studentu, a potom vrši: a) Upis semestra na datum upisa studenta. b) Dodjelu predmeta odabrano nastavnog plana na upisanoj godini studenta. c) Generisanje pristupnih podataka (broj indeksa i lozinka). 5. Sistem vraća referenta na pretragu matične knjige studenata sa označenim podacima o upisanom studentu.
Alternativni tok(ovi) događaja:	1a. Referent bira opciju za import podataka o studentu putem excel dokumenta, te od sistema dobiva predefinisani template koji je neophodno popuniti. Upis studenta se vrši postavljanjem ispravno popunjenog excel dokumenta na sistem.
Exeption(s) flow:	3a. Ukoliko podaci uneseni u sklopu prvog koraka ne zadovoljavaju validacijska pravila, sistem informiše referenta o potrebnim ispravkama i onemogućava mu prelaz na naredni korak (dupla ili pogrešna mail adresa, neodgovarajući format telefona...). 3b. Uneseni broj indeksa je zauzet i nije moguće kompletirati upis studenta niti preći na naredni korak.



Zadatak 2 (za vježbu): eStudent

Modul *eStudent* je namijenjen svim upisanim studentima na određenom univerzitetu/fakultetu. Modul je zatvorenog tipa, što znači da je za pristup njegovim mogućnostima potreban validan korisnički račun, odnosno uspješna prijava. U sklopu prijave na sistem, softver nudi mogućnost resetovanja lozinke u slučaju da je ista zaboravljena. Za uspješan reset lozinke student treba unijeti validnu mail adresu na koju sistem prosljeđuje novu lozinku za prijavu. Također, ručnu izmjenu lozinke je moguće uraditi i naknadno.
Po uspješnoj prijavi, student dobiva uvid u posljednje aktivnosti na predmetima, preciznije uvid u posljednja obavještenja, posljednje objavljene nastavne sadržaje i dokumente, kao i prikaz notifikacija u vezi sa prethodno obavljenim aktivnostima na sistemu. Pored toga, student ima mogućnost detaljnije pretrage arhive obavještenja, nastavnih sadržaja i dokumenata, te pregleda detalja uz eventualno preuzimanje priloženih dokumenata.
Jedna od najvažnijih mogućnosti koju sistem nudi studentima jeste prikaz tzv. *eIndex-a*. *eIndex* je zapravo elektronska verzija svih značajnih podataka o studiju određenog studenta. On nudi mogućnost uvida u statusne podatke studenta, pregled položenih i nepoloženih ispita uz opcionalni prikaz detalja o evaluciji na predmetu (način formiranja konačne ocjene), pregled kompletnog nastavnog plana i programa, pregled uplata i upisanih semestara. Također, studenti imaju uvid u svoje lične i kontakt podatke, gdje mogu poslati i zahtjev za promjenom podataka ukoliko uoče određene greške.
Sistem studentima pruža i elektronsku prijavu ispita, kao i detaljan pregled objavljenog rasporeda ispita. Student može prijaviti više ispita odjedanput, kao i odjaviti prethodno prijavljene ispite.
U konačnici, modul *eStudent* podržava i slanje zahtjeva za uvjerenjima, te pregled svih prethodno procesiranih zahtjeva. Zahtjev može biti uspješno ili neuspješno procesiran, pri čemu se kod neuspješnog procesiranja studentima daje na uvid i razlog zbog kojeg uvjerenje nije bilo moguće izdati.
Diplomirani studenti također imaju mogućnost pristupa sistemu, gdje im se nakon diplomiranja, na uvid daju detalji o diplomskom radu (podaci o temi diplomskog rada i ocjeni diplomiranja, podaci o mentoru i izvještaju komisije sa odbrane diplomskog rada). Pored toga, svim diplomiranim studentima se otvara pristup tzv. alumni zajednici gdje im je omogućena komunikacija sa poslodavcima.

a) **Use-case dijagramom modelirati komponentu softvera eUniverzitet namijenjenu studentima.**
b) **Uraditi detaljnu specifikaciju slučaja korištenja „Prijava ispita".**

Ispitni zadatak 1:

Određena aviokompanija je izdala specifikaciju zahtjeva za implementaciju softverskog rješenja koje treba da prvenstveno podrži upravljanje letovima, te rezervaciju karata kao osnovne aspekte poslovanja.

Upravitelj letova unosi podatke o zaposlenom osoblju (*Ime, Prezime, Datum zaposlenja, Adresa, Email, CV*) aviokompanije. Svako od članova osoblja ima određen tip posla koji obavlja. Upravitelj se brine i za evidenciju podataka o letu (*Datum polaska, Vrijeme*

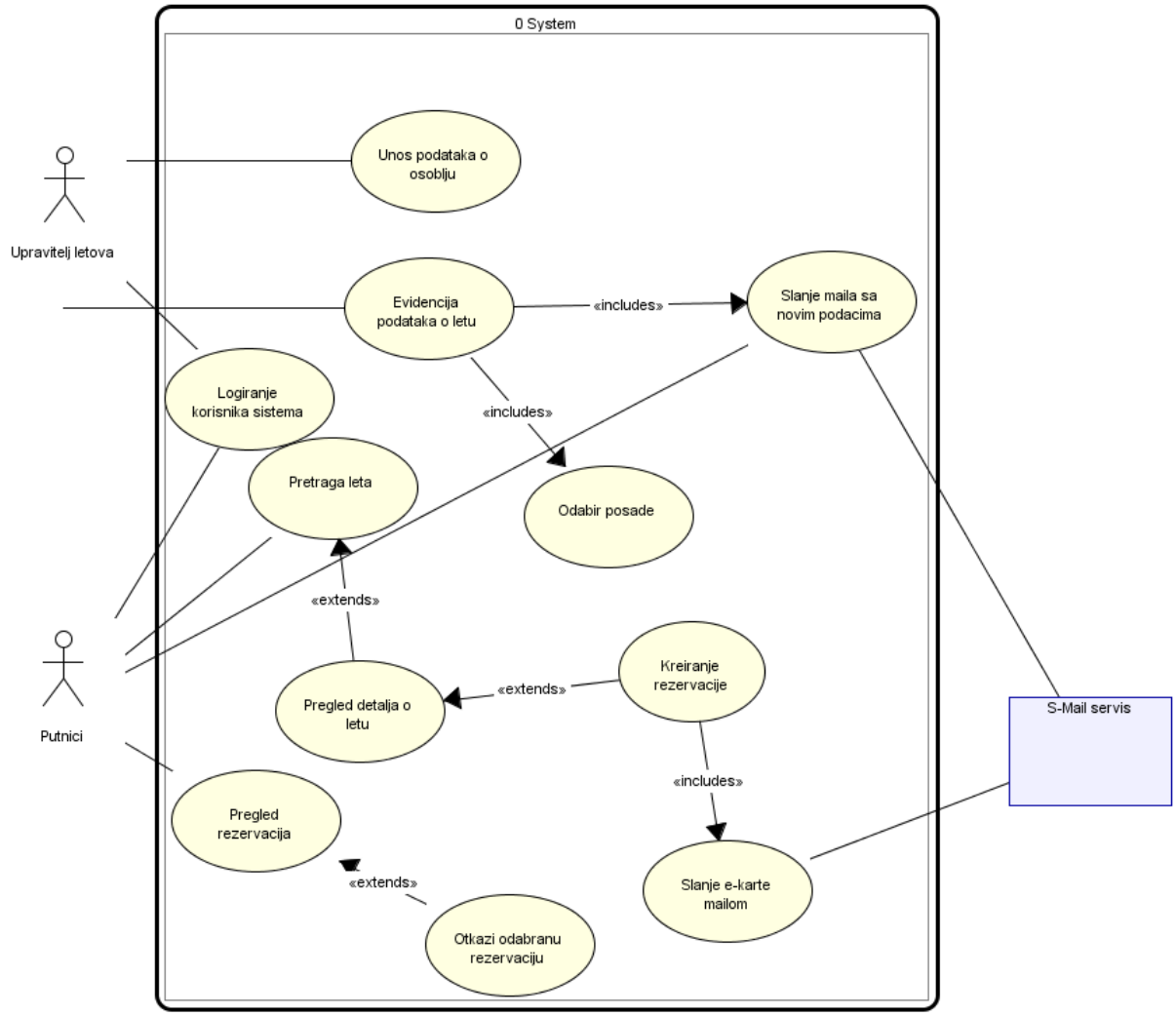
polaska, Trajanje, Polazište, Odredište, Broj sjedišta, Cijena), što uključuje i odabir kompletne posade za let. Posadu čini zaposleno osoblje aviokompanije.

Putnici (*Broj pasoša, Ime, Prezime, Email, Lozinka*) imaju mogućnost pretrage letova po destinaciji i periodu u kojem je let zakazan. Pregledom detalja o letu dobivaju mogućnost kreiranje rezervacije (*Broj karata, Klasa leta, Datum, Cijena*) za odabrani let. Rezervaciju mogu napraviti samo registrovani putnici. Cijena rezervacije se formira na osnovu početne cijene leta, broja karata i odabrane klase leta (ekonomska klasa ne mijenja cijenu leta, dok prva klasa osnovnu cijenu leta uvećava za 10%). Rezervaciju nije moguće zaključiti ukoliko nema dovoljno slobodnih sjedišta na letu. Nakon uspješnog zaključivanja rezervacije putnik dobiva e-kartu na mail.

Svakom promjenom podataka o letu sistem svim putnicima tog leta šalje mail sa izmijenjenim podacima. Putnici putem pregleda rezervacija uvijek mogu i otkazati odabranu rezervaciju.

Rješenje studenta:

Ispod je prijedlog rješenja studenta koji je na ispitu osvojio maksimalan broj bodova (20 od mogućih 20). Dijagram je kreiran u alatu Open ModelSphere.



Ispitni zadatak 2:

Za kompaniju "XY" potrebno je izradom UML dijagrama dati prijedlog softverskog projekta za podršku procesa upravljanja softverskim zahtjevima. Kompanija se primarno bavi razvojem softvera za druge softverske kompanije (Naziv, Adresa, Kontakt, Web).

Jedan softverski projekat (Naziv, Opis, Rok, Zavrsen) čine faze projekta, gdje se svakoj fazi pridružuje skup zahtjeva. Upravljanje fazama projekta i definisanje zahtjeva projekta je zadatak voditelja razvojnog tima (Ime, Prezime, e-mail, Lozinka, Certfikat, GodineIskustva). U procesu definisanja zahtjeva također učestvuje i klijent (kompanija) kome je softversko rješenje namijenjeno. Svaka izmjena zahtjeva na projektu zahtijeva potvrdu klijenta kako bi on bio finalan. Klijenti mogu pratiti tok izvršenja pojedinih faza projekta. Voditelj razvojnog tima također brine o raspodjeli zahtjeva među članovima tima (Ime, Prezime, Email, Lozinka, Pozicija, BrojAktivnihZadataka), odnosno dodjeli konkretnih zadataka. Pri svakoj dodjeli zahtjeva određenom članu tima voditelj dodaje napomenu koja dodatno pojašnjava dobiveni zadatak, kao i rok do kojeg se zahtjev mora kompletirati i eventualno prilog. Na jednom zahtjevu može da radi više članova tima. Jedan član tima ne može raditi na više od 3 zahtjeva istovremeno.

Članovi razvojnog tima dobivaju listu svojih zadataka, odnosno aktivnih zahtjeva na kojima trenutno rade, gdje je moguće izmijeniti trenutni status zadatka (npr. u izradi ili završen). Članovi tima također mogu razmijenjivati poruke putem sistema.

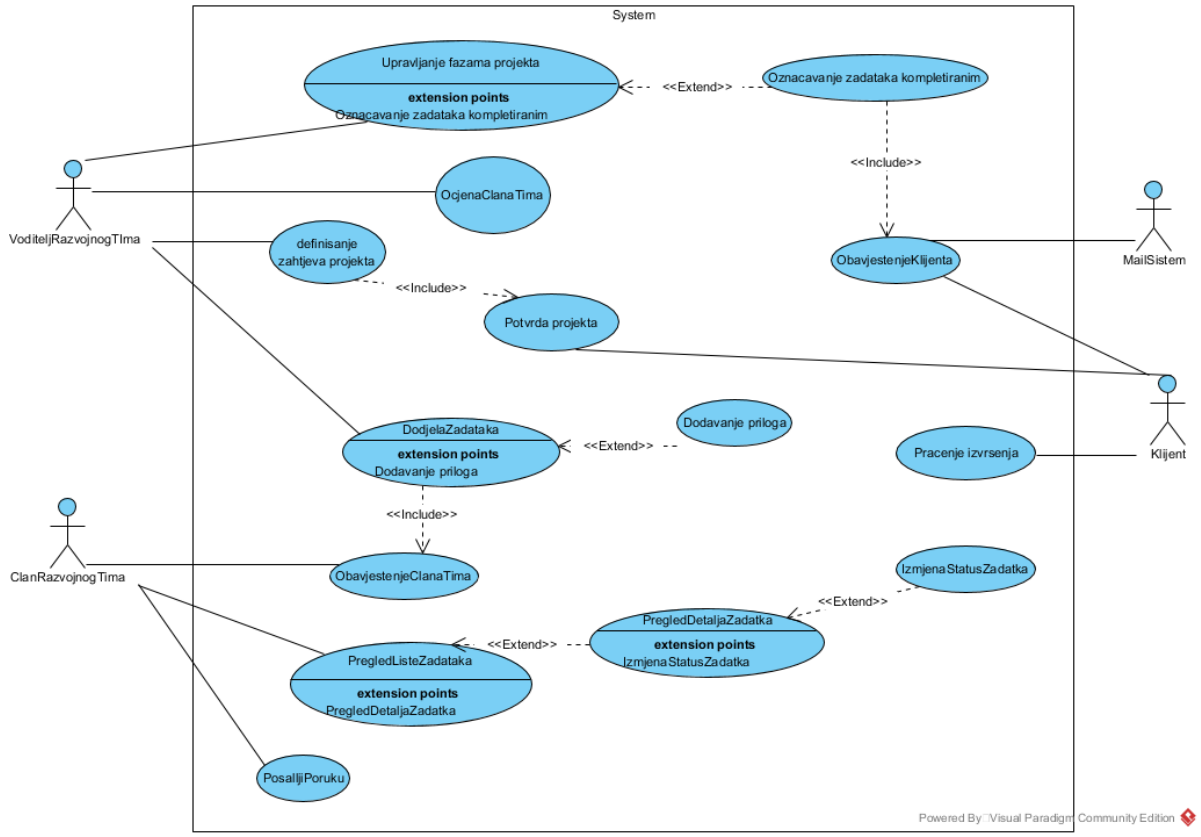
Voditelj ocjenjuje (numerički i opisno) sve članove tima na određenom softverskom projektu, te obavještava klijenta putem mail-a i sistema kada su svi zahtjevi kompletirani.

a) **(25 bodova) Use-case dijagramom modelirati funkcionalnosti opisanog softverskog rješenja.**
b) **(5 bodova) Kreirati use-case dokument za slučaj korištenja „Ocjena članova tima".**

Rješenje studenta:

Ispod je prijedlog rješenja studenta koji je na ispitu osvojio maksimalan broj bodova za dijagram (25b od mogućih 25b), te maksimalan broj bodova i za use-case dokument (5b od mogućih 5b). Dijagram je kreiran u alatu Visual Paradigm. Izdvojeno rješenje sa ispita svakako ima i nedostataka, međutim kada se posmatra cjelokupno razumijevanje use-case modela i njegovih komponenti rješenje je ocijenjeno sa maksimalnim brojem bodova.

a) Use-case dijagram



b) Use-case dokument

Preduslovi: Voditelj mora biti prijavljen na sistem. Projekat, tj. svi zahtjevi moraju biti u potpunosti implementirani.

Rezultati: Sistem u bazu podataka uspješno pohranjuje podatke o tome koji je voditelj ocijenio kojeg člana tima i bilježi datum ocjenjivanja i ocjenu.

Glavni tok:

1) Voditelj se prijavljuje na sistem i otvara formu za odabir projekata.
2) Voditelj bira projekat koji želi pregledati i nakon toga potvrđuje odabir.
3) Nakon toga bira opciju Ocjenjivanja članova tima.
4) Sistem vraća listu članova tima koji su radili na (odabranom) projektu.
5) Voditelj unosi ocjenu, eventualno napomenu. Nakon toga potvrđuje unos ocjene.
6) Sistem sprema podatke o ocjeni i obavještava člana tima.

Exceptions:

1a) Korisnik unio pogrešne podatke i sistem ga upozorava na grešku i traži ponavljanje prijave.

5a) Voditelj ne može ocjeniti člana tima jer svi zahtjevi nisu u potpunosti implementirani. Sistem vraća listu zahtjeva koji nisu implementirani.

Prateće video lekcije možete pogledati u sklopu *YouTube* kanala na sljedećim linkovima:

- Use-case model (Visual Paradigm 14.0)
- Use-case dijagram (Open ModelSphere 3.2)
- Use-case dokument (Open ModelSphere 3.2)
- Use-case dijagram ispitni zadatak (Open ModelSphere 3.2)

Dodatni primjer: Prijem pacijenata - Hospital Reception (Dostupno: 06.03.2018.).

3. Literatura

1. Martin Fowler, UML Distilled Third Edition: A brief guide to the standard object modeling language, 2004. Download (Dostupno: 06.03.2017.).
2. Craig Larman, Applyng UML and paterns, Prentice Hall, 2004.
3. https://www.visual-paradigm.com/
4. https://www.youtube.com/user/VisualParadigm/
5. http://www.uml.org/