SOFTVERSKO INŽENJERSTVO

školska 2024/2025 godina

Vežba 4: Use Case dijagrami

Use Case dijagrami (tzv. dijagrami slučajeva upotrebe) predstavljaju jedan od najvažnijih dijagrama u **UML-u** (Unified Modeling Language). Oni su često prvi dijagrami koji se izrađuju u fazi planiranja nekog sistema, pre klasnih i sekvencijalnih dijagrama koji dolaze posle.

Njihova glavna svrha je da **prikažu kako korisnici interaguju sa sistemom**, odnosno da odgovore na pitanje: **Šta korisnik želi da postigne korišćenjem sistema?**

Za razliku od sekvencijalnih ili klasnih dijagrama koji se bave **strukturama i ponašanjem unutar sistema**, Use Case dijagrami prikazuju **funkcionalnosti iz spoljnje perspektive**, tj. iz ugla **korisnika** i **njihovih ciljeva**.

Takav pristup je naročito koristan prilikom sastanaka sa klijentima, jer omogućuje svima da razumeju opseg sistema i diskutuju o funkcionalnostima bez potrebe za stručnim znanjem.

🎯Osnovni ciljevi:

1. **Prikaz osnovnih funkcionalnosti sistema**
   Use Case dijagrami daju **vizuelni pregled** šta sistem sve može da radi i koje su njegove osnovne funkcionalnosti.
2. **Identifikacija korisnika (aktera) i njihove interakcije sa sistemom**
   Omogućavaju da razumemo **ko koristi sistem** i **kako ga koristi**.
3. **Jednostavna komunikacija između timova**
   Dijagrami su razumljivi i programerima i klijentima koji možda nemaju tehničko znanje.
4. **Upotreba u fazi analize i dizajna softvera**
   Veoma su korisni u ranoj fazi razvoja jer pomažu u **identifikaciji zahteva korisnika** i služe kao osnova za kreiranje drugih dijagrama (npr. sekvencijalnih).

👤 Akteri (Actors)

U okviru Use Case dijagrama, **akteri** su svi entiteti **koji imaju neku vrstu interakcije sa sistemom**. Akteri iniciraju funkcionalnosti sistema, tj. pokreću slučajeve upotrebe.

Oni *ne moraju* uvek da budu ljudi – akteri mogu biti:

- fizičke osobe (npr. student, administrator, prodavac),
- drugi softverski sistemi (npr. bankovni sistem koji proverava plaćanja),
- uređaji (npr. skener kartice, printer).

Akteri nam pomažu da shvatimo **ko koristi sistem** i **zašto ga koristi**. Razumevanje aktera vodi ka boljem definisanju zahteva i funkcionalnosti koje sistem treba da pruži.

📂 **Vrste aktera:**

1. Primarni akteri

To su entiteti **koji imaju direktan cilj** koji žele da ostvare korišćenjem sistema. Na primer:

- **Student** koji želi da prijavi ispit
- **Kupac** koji želi da naruči proizvod
- **Administrator** koji želi da doda novog korisnika

👉 Oni su često inicijatori slučajeva upotrebe – sistem postoji da bi ispunio njihove potrebe.

2. Sekundarni akteri

Oni **ne koriste sistem direktno**, ali omogućavaju da sistem funkcioniše pravilno. To su:

- **Baza podataka** koja čuva informacije
- **Spoljni e-mail servis** koji šalje notifikacije korisnicima
- **Autentikacioni sistem** koji proverava lozinke

👉 Oni su „iza scene“, ali bez njih sistem ne bi mogao da obavi funkcionalnosti koje primarni akteri zahtevaju.

Nakon što identifikujemo aktere, sledeći korak je da definišemo **šta tačno žele da postignu kroz interakciju sa sistemom**.

🎬 Slučajevi upotrebe

Slučajevi upotrebe (Use Cases) predstavljaju osnovne funkcionalnosti koje sistem pruža svojim korisnicima – tj. *šta* korisnik želi da postigne korišćenjem sistema. Oni odgovaraju na pitanje: **"Koje akcije korisnik može da preduzme pomoću sistema?"**

U dijagramu se svaki slučaj upotrebe prikazuje kao **elipsa**, i povezan je sa jednim ili više aktera koji taj slučaj iniciraju.

Slučajevi upotrebe se fokusiraju na **funkcionalne zahteve sistema**, bez ulaska u tehničke detalje implementacije. Na primer:

- „Prijavi se“ – korisnik želi da pristupi svom nalogu
- „Dodaj artikal“ – administrator dodaje proizvod u katalog
- „Pošalji izveštaj“ – sistem šalje mesečni izveštaj rukovodiocu
- „Generiši potvrdu“ – student generiše PDF potvrdu o upisanom predmetu

Ove elipse predstavljaju konkretne aktivnosti koje korisnici žele da sprovedu kako bi ostvarili određeni cilj. Jedan akter može inicirati više slučajeva upotrebe, a jedan slučaj upotrebe može biti zajednički za više aktera.

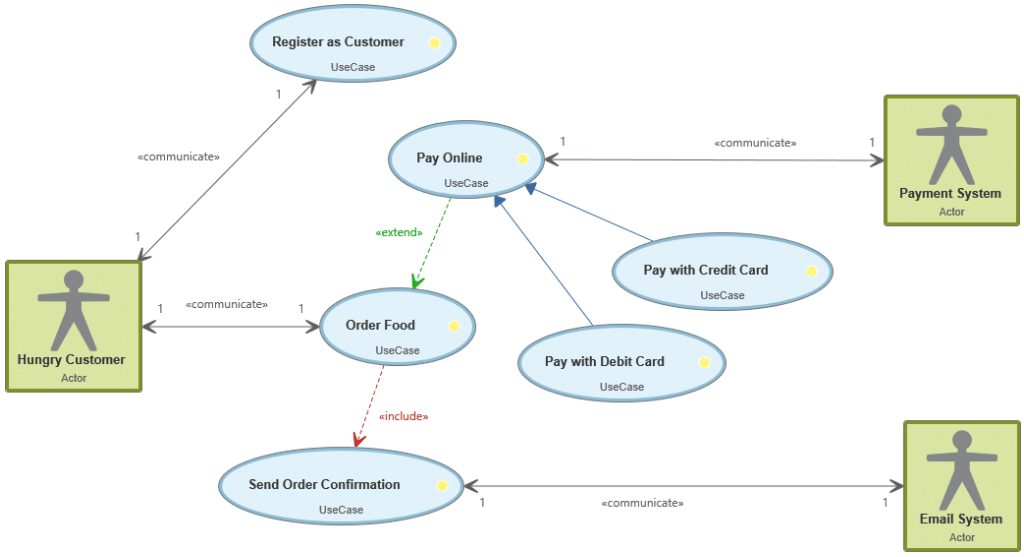
Svaki slučaj upotrebe treba da:

- **Ima jasan i merljiv cilj**

  Akter koristi sistem da bi nešto postigao (npr. završio narudžbinu, izmenio podatke, itd.).
- **Bude opisan jasnom glagolskom frazom**

  Npr. „Dodaj korisnika“, „Pregledaj naloge“, „Pošalji zahtev“.
- **Bude pokrenut od strane jednog ili više aktera**

  Bilo da je to fizička osoba, drugi sistem, aplikacija, itd.



✅ **Pravila dobre prakse**

1. Imenuj slučajeve glagolskim frazama

- Imena treba jasno da izražavaju akciju i cilj (npr. *"Kreiraj izveštaj"*, *"Izmeni lozinku"*, a ne samo *"Izveštaj"*, *"Lozinka"*).

2. Aktere uvek smeštaj van sistema (izvan pravougaonika)

- Sistem je predstavljen pravougaonikom, a sve interakcije dolaze spolja, preko aktera.

3. Ne ulazi u tehničke detalje

- Dijagram treba da pokaže *šta* se radi, a ne *kako*. Tehničke implementacije se opisuju u drugim dijagramima, poput sekvencijalnih ili dijagrama aktivnosti.

4. Fokusiraj se na korisničku perspektivu

- Dijagram treba da bude razumljiv svima, prvenstveno krajnjim korisnicima.

5. Čuvaj preglednost

- Dijagram ne treba da bude prenatrpan – bolje je podeliti na više manjih dijagrama po funkcionalnim celinama nego imati jedan ogroman i nepregledan.

🔁 Veze između elemenata

Veze u Use Case dijagramima predstavljaju način na koji su elementi (akteri i slučajevi upotrebe) međusobno povezani. One omogućavaju modeliranje kompleksnijih odnosa između funkcionalnosti i korisnika sistema.

1. Association (Asocijacija)

Ovo je najosnovnija veza koja se prikazuje **linijom** između aktera i slučaja upotrebe. Označava da **akter koristi određenu funkcionalnost** sistema.

📌 **Primer**:
Student ——> (Prijavi ispit)
Ova linija pokazuje da student može da koristi funkcionalnost „Prijavi ispit“.

2. Include (Uključi)

Koristi se kada jedan slučaj upotrebe **obavezno uključuje** drugi, svaki put kada se izvršava. Označava da je neki Use Case **neodvojiv** deo drugog i **uvek se poziva** u okviru njega. Prikazuje se isprekidanom linijom sa oznakom <<include>>.

📌 **Primer**:
(Kupi knjigu) ⟶<<include>>⟶ (Unesi adresu)
Svaki put kada korisnik kupuje knjigu, mora uneti adresu – to je obavezna funkcionalnost.

🧠 Koristi se kada želimo da izbegnemo dupliranje koda ili opisa u više Use Case-ova koji pozivaju istu funkcionalnost.

3. Extend (Proširi)

Koristi se kada jedan Use Case **može, ali ne mora** da koristi drugi, zavisno od određenih uslova. Označava **opcioni dodatak** funkcionalnosti, koji se prikazuje isprekidanom linijom sa oznakom <<extend>>.

📌 **Primer**:
(Prijavi se) ⟶<<extend>>⟶ (Dvofaktorska autentifikacija)
U nekim slučajevima, kada je aktivirana sigurnosna opcija, korisnik mora da prođe dodatnu autentifikaciju.

🧠 Koristi se za prikazivanje alternativnih ili uslovnih tokova ponašanja koji nisu uvek potrebni.

4. Generalizacija (Nasleđivanje)

Veza koja pokazuje da **jedan akter ili slučaj upotrebe nasleđuje ponašanje** drugog. Prikazuje se **linijom sa praznom strelicom** ka roditeljskom elementu. Omogućuje **ponovno korišćenje funkcionalnosti** i bolje organizovanje hijerarhije.

📌 **Primer – Akteri**:
(Administrator) —▷ (Korisnik)
Administrator nasleđuje sve funkcionalnosti običnog korisnika, ali ima i dodatne opcije.

📌 **Primer – Use Case-ovi**:
(Upravljaj korisnicima) —▷ (Izmeni korisnika)
„Izmeni korisnika“ je specijalizovan Use Case koji koristi osnovne osobine opšteg slučaja „Upravljaj korisnicima“.

🧠 Generalizacija pomaže u organizaciji složenih sistema i izbegavanju dupliranja sličnih funkcionalnosti.

🛠️ Praktična primena:

Use Case dijagrami nisu samo teorijski alat – oni imaju široku i **konkretno primenljivu ulogu** u različitim fazama razvoja softverskog sistema. Evo kako se koriste u praksi:

1. Definisanje korisničkih zahteva

Use Case dijagrami pomažu timu da **jasno identifikuje šta korisnici očekuju od sistema**. Umesto tehničkih specifikacija, koristi se **prirodan jezik** i vizuelni prikaz koji je lako razumljiv i tehničkim i netehničkim licima.

✅ *Na primer:* Klijent kaže da želi da korisnici mogu da „prijave ispit“ i „provere rezultate“. Ove funkcionalnosti se direktno prikazuju kao Use Case-ovi u dijagramu.

2. Analiza i dizajn softverskog sistema

Na osnovu Use Case dijagrama, analitičari i dizajneri **razbijaju funkcionalnosti na manje celine**, analiziraju tokove rada i planiraju dalju **unutrašnju arhitekturu sistema**. Use Case dijagrami često služe kao **polazna tačka za kreiranje drugih UML dijagrama**.

3. Pisanje test scenarija (Test Case)

Svaki Use Case se lako može pretvoriti u **test scenario** koji proverava da li sistem ispravno izvršava određenu funkcionalnost. To pomaže testerima da obuhvate sve funkcionalnosti koje su korisnicima važne.

✅ *Na primer:* Ako postoji Use Case „Resetuj lozinku“, test slučaj će proveriti da li korisnik može da unese email, dobije kod i resetuje lozinku.

4. Dokumentacija projekta

Use Case dijagrami predstavljaju **važan deo projektne dokumentacije** jer na pregledan način pokazuju **šta sistem radi** i **ko koristi koju funkcionalnost**. Idealni su za dodavanje u tehničke i netehničke delove dokumentacije jer su vizuelno (a i deskriptivno) kratki i jasni.

5. Komunikacija između tima i klijenta

Pošto ne zahtevaju tehničko predznanje, Use Case dijagrami su **efikasan alat za diskusiju sa klijentima** i korisnicima. Omogućuju da svi učesnici projekta budu **na istoj strani**, jer svi vide iste funkcionalnosti i mogu komentarisati svoje potrebe i očekivanja.

🧠 *Dodatni benefit:* Mogu se koristiti i tokom prezentacija, sastanaka ili demonstracija koncepta (proof of concept), da bi se lakše objasnilo kako će sistem raditi.

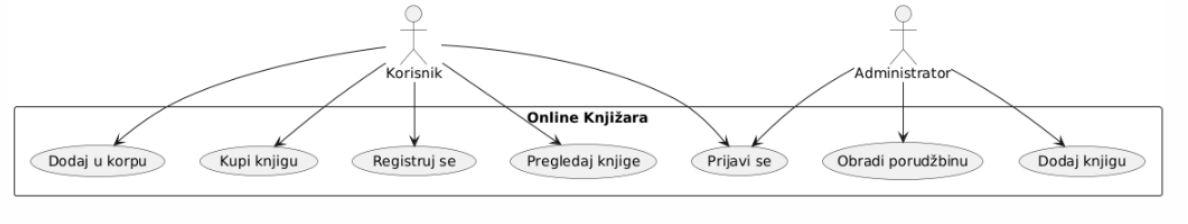
📘 Primer 1: Online knjižara

🎭 Akteri:

- Korisnik (kupuje knjige)
- Administrator (dodaje nove knjige, obrađuje porudžbine)

🎬 Slučajevi upotrebe:

- Registruj se
- Prijavi se
- Pregledaj knjige
- Dodaj u korpu
- Kupi knjigu
- Dodaj knjigu
- Obradi porudžbinu



U sistemu online knjižare, korisnik ima mogućnost da se registruje, prijavi, pretražuje i dodaje knjige u korpu, a zatim ih i kupi. Administrator ima dodatne privilegije – može unositi nove knjige i obrađivati porudžbine koje stignu od korisnika. Ovim dijagramom jasno je prikazana interakcija između aktera i funkcionalnosti sistema.

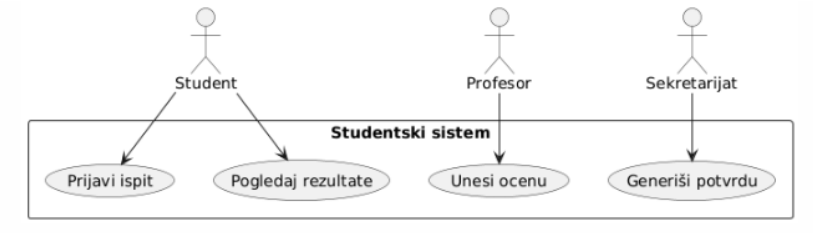
📘 Primer 2: Studentski informacioni sistem

🎭 Akteri:

- Student
- Profesor
- Sekretarijat

🎬 Slučajevi upotrebe:

- Prijavi ispit
- Pogledaj rezultate
- Unesi ocenu
- Generiši potvrdu



Studentski informacioni sistem omogućava studentima da prijave ispite i vide svoje ocene. Profesori unose ocene ispita, a službenici iz sekretarijata mogu generisati razne studentske potvrde (npr. potvrdu o redovnom studiranju). Ovaj sistem jasno odvaja funkcionalnosti po vrstama korisnika.