MRS

LAB 05

MRSLab05 – Metodologija Razvoja Softvera – Vežba 05

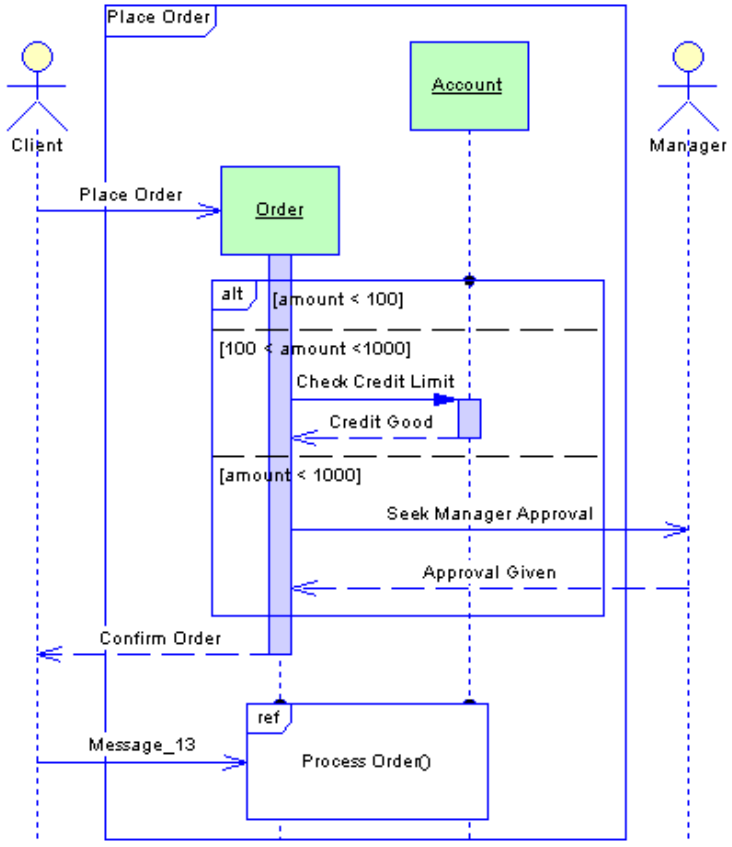
Dijagram senkvenci
Komunikacioni dijagram

1. Dijagram sekvenci

Dijagram sekvenci je UML dijagram interakcije. On prikazuje hronologiju prenošenja poruka između objekata u sistemu i učesnika. Mogu se koristiti da ilustruju mogući scenario načina upotrebe, izvršenje operacija, ili jednostavno interakciju između klasa sistema.

Može se koristiti jedan ili više sekvencijalnih dijagrama da bi se odredili načini upotrebe, ili da bi se identifikovale sve mogućnosti složenog ponašanja sistema.

Jedna od glavnih prednosti dijagrama sekvenci predstavlja to što se mogu prikazati najčešće interakcije, a takođe se lako mogu dodati alternativni ili paralelni scenariji korišćenjem interakcionih fragmenata. Zbog toga, u jednom dijagramu sekvence se može opisati više povezanih interakcija.



Slika broj 1. Primer dijagrama sekvenci

U primeru gore, učesnik Client šalje narudžbinu. Poruka *Place Order* kreira objekat *Order*. Fragment interakcije obrađuje različite mogućnosti za potvrdu narudžbine. Objekat *Account* i učesnik *Manager* mogu vršiti interakciju sa narudžbinom u zavisnosti od veličine narudžbine. Kada

se pošalje poruka *Confirm order* započeta je interakcija *Process Order*. Interakcija se nalazu u drugom dijagramu sekvenci, a ovde je predstavljena preko reference interakcije.

1.1. Analiza slučajeva korišćenja

Dijagram sekvenci se može koristiti da bolje objasni slučajeve korišćenja sistema. Ovaj pristup je koristan kod analize zahteva jer može pomoći kod identifikacije klasa i asocijacija koje nisu uočene na početku.

Često je potrebno kreirati više dijagrama kako bi se opisali svi mogući scenariji slučajeva korišćenja. U ovakvim situacijama, dobro je koristiti dijagrame sekvenci kako bi se otkrili svi potrebni objekti pre identifikovanja klasa koje ih instanciraju. Posle identifikovanja klasa, mogu se zaključiti asocijacije između njih na osnovu poruka koje se šalju između objekata.

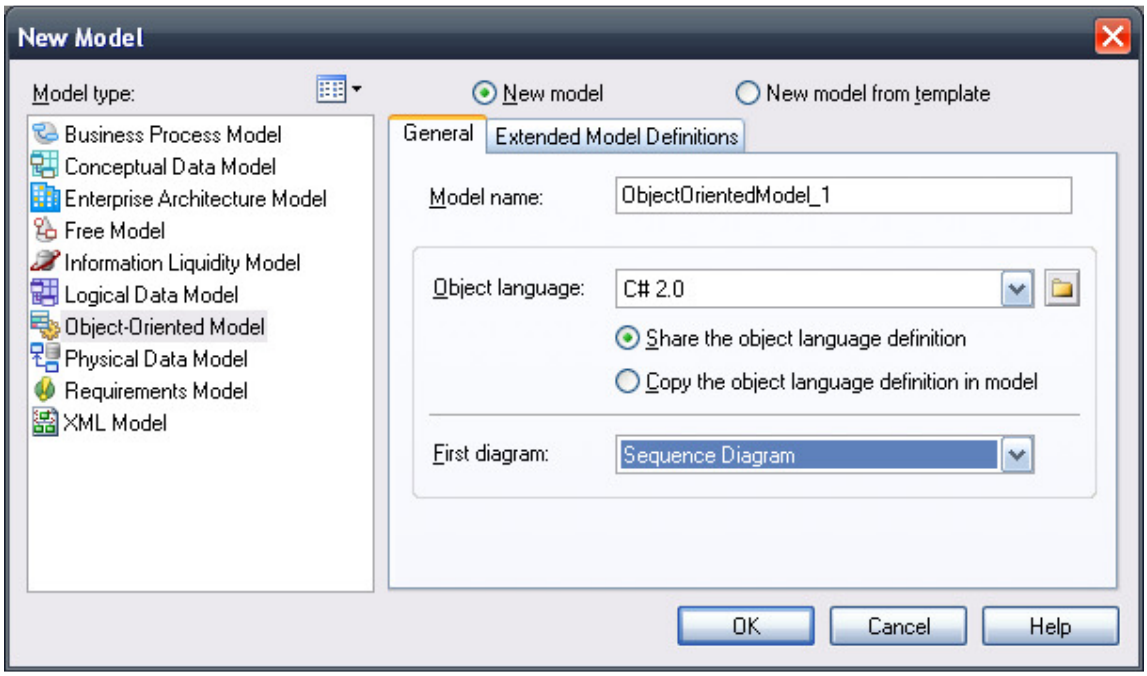
1.2. Analiza dijagrama klasa

Kreiranje dijagrama sekvenci može biti mogućnost da bi se testirao statički model na nivou koncepta: on može predstavljati scenario u kom su klase iz klasnih dijagrama instancirane kako bi se kreirali objekti neophodni za pokretanje scenarija.

On dopunjuje dijagram klasa koje predstavlja statičku strukturu sistema tako što definiše ponašanje klasa, interfejsa i moguću upotrebu njihovih operacija.

1.3. Kreiranje dijagrama sekvenci

Da bi kreirali novi dijagram sekvenci iz menija u PowerDesigner-u biramo File → New Model da bi se prikazao prozor New Model kao na sledećoj slici.

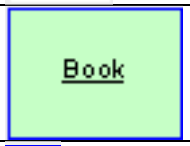
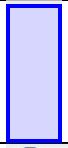
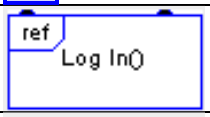
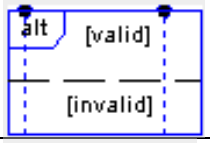
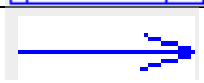
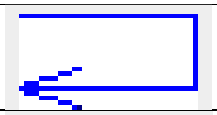
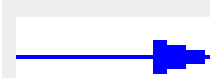
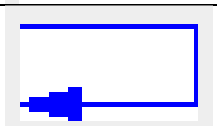
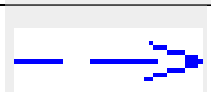


Slika broj 2. Kreiranje dijagrama sekvenci

U listi tipova modela na levoj strani prozora biramo "Object-Oriented Model". U polju First diagram biramo "Sequence Diagram".

2. Objekti u dijagramu sekvenci

U dijagramu sekvenci se najčešće kreiraju sledeći objekti:

Objekat	Alat	Simbol	Opis
Učesnik			Spoljašnja osoba, proces ili nešto što vrši interakciju sa sistemom, podsistemom ili klasom.
Objekat			Predstavlja klasu.
Aktivacija			Vreme potrebno da se izvrši procedura ili funkcionalnost.
Referenca interakcije			Referenca na drugi dijagram sekvenci.
Fragment interakcije			Kolekcija povezanih poruka.
Poruka			Komunikacija koja prenosi informacije
Rekurzivna poruka			Rekurzivne poruke: objekat koji šalje je i objekat koji prima.
Poruka poziva procedure			Poruka poziva procedure sa podrazumevanom aktivacijom.
Samopozivajuća poruka			Rekurzivna poruka poziva procedure sa podrazumevanom aktivacijom.
Povratna poruka			Specificira kraj procedure. Najčešće je povezana sa pozivom procedure, povratna poruka može biti izostavljena jer se podrazumeva na kraju aktivacije.
Rekurzivna povratna poruka			Rekurzivna poruka sa povratkom toka kontrole.

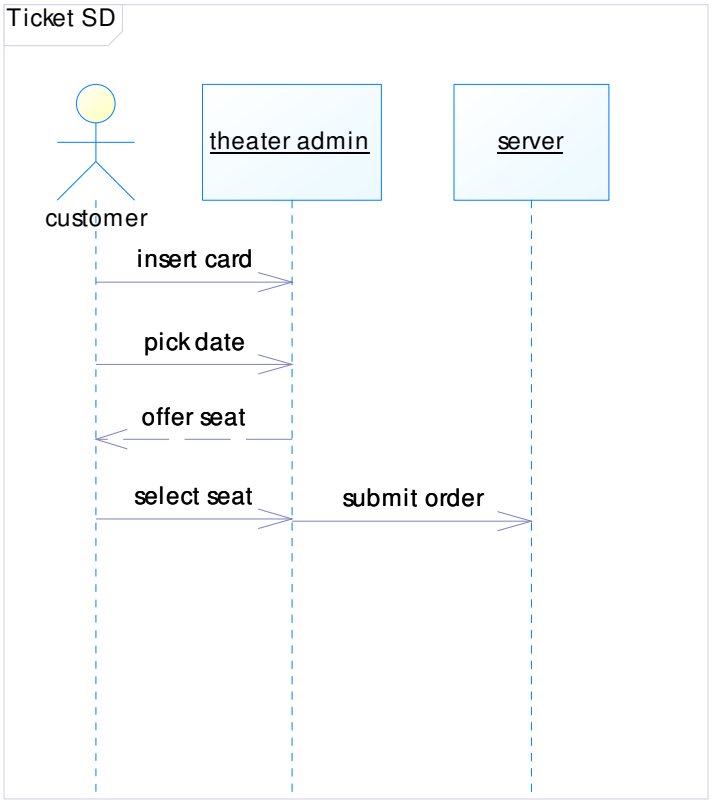
Slika broj 3. Objekti dijagrama sekvenci

2.1. Učesnik

Učesnik je spoljašnji korisnik ili grupa korisnika koji vrše interakciju sa sistemom. Učesnici mogu biti ljudi ili drugi spoljašnji sistemi. Učesnici su uglavnom oni entiteti čije ponašanje ne možemo da kontrolišemo ili menjamo, jer oni nisu deo sistema koji opisujemo.

U dijagramima sekvenci učesnik ima liniju života koja predstavlja vreme njegovog života. Učesnik se ne može razdvojiti od svoje linije života.

Ukoliko je učesnik taj koji započinje interakciju, on je obično predstavljen prvom (skroz levom) linijom života u dijagramu sekvenci. Ukoliko imamo nekoliko učesnika u dijagramu, treba pokušati da se oni smeste tako da predstavljaju skroz leve ili skroz desne linije života jer su učesnici po svojoj definiciji spoljašnji učesnici sistema.



Slika broj 4. Primer učesika

2.2. Objekat

Objekat se definiše kao deo sistema koji se opisuje. Mogu se predstaviti tri moguće situacije vezane za prikaz objekta:

- objekat nije instanca klase - on ima samo ime
- objekat predstavlja instancu klase - ima ime i klasu
- objekat predstavlja instancu klase ali zapravo predstavlja bilo koju instancu klase - ima klasu ali nema ime.

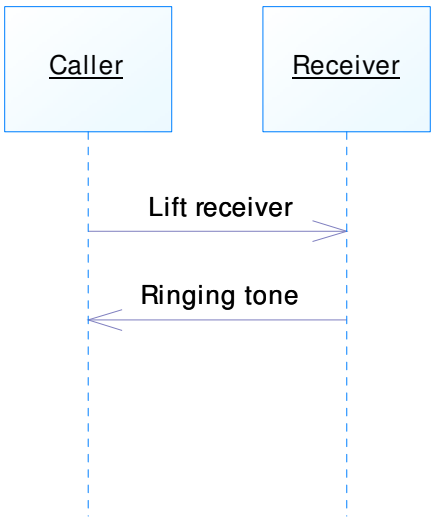
U dijagramima sekvenci objekat ima liniju života: to je isprekidana vertikalna linija koja se nalazi ispod simbola za objekat. Prolazak vremena se na stranici uvek prikazuje na dole. Linija

života objekta prikazuje period tokom kojeg objekat postoji. Objekat se ne može odvojiti od svoje linije života.

Ukoliko je objekat kreiran ili uništen tokom vremenskog perioda koji se prikazuje na dijagramu, onda njegova linija života počinje ili se završava u odgovarajućoj tački.

Objekti se pojavljuju na vrhu dijagrama. Oni među sobom razmenjuju poruke.

Objekat koji postoji kada razmena poruka počne je prikazan na vrhu dijagrama, iznad prve strelice koja predstavlja poruku. Linija života objekta koji i dalje postoji kada je transakcija završena, se nastavlja ispod poslednje strelice poruke.



Slika broj 5. Objekti i linije života (timeline)

2.3. Aktivacija

Aktivacije predstavljaju opcione simbole koji prikazuju vreme potrebno da se neka akcija izvrši. Oni se kreiraju na liniji života objekta. Oni su čisti simboli i nemaju svoju stranicu sa osobinama.

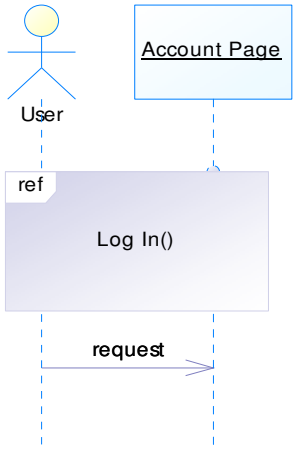
Aktivaciji se mogu pridružiti ili se sa nje mogu ukloniti poruke. Takođe je moguće promeniti veličinu aktivacije, pomerati je ili prouzrokovati da aktivacija preklopi druge aktivacije.

Aktivacija se automatski kreira kada se kreira poruka poziva procedure.

2.4. Referenca interakcije

Referenca interakcije se koristi da predstavi jedan dijagram sekvenci koji se nalazi u telu drugog. Ovo osobina omogućava da se često korišćene interakcije grupišu u module i više puta koriste u nizu dijagrama sekvence.

U sledećem primeru korisnik mora da se loguje na sistem pre nego što bude u mogućnosti da pristupi stranici sa računima na web aplikaciji. Pošto će se proces logovanja pojavljivati kao deo velikog broja interakcija sa web aplikacijom, on je izdvojen u poseban dijagram sekvenci koji se naziva *Log In*, i predstavljen je pomoću reference interakcije:



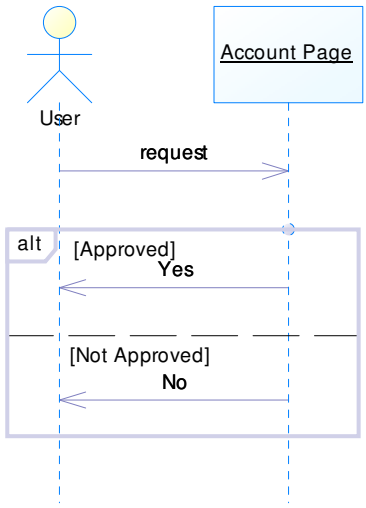
Slika broj 6. Referenca interakcije

Referenca interakcija se može pomerati, može joj se menjati veličina, itd. Kada simbol preklopi liniju života objekta, on se povezuje sa objektom automatski, a ova povezanost je prikazana sa malim ispupčenjem na gornjoj strani simbola gde se on susreće sa linijom života. Ukoliko pomerimo simbol od linije života objekta, on automatski prestaje da bude povezan.

2.5. Fragment interakcije

Fragment interakcije omogućava da se grupišu povezane poruke u dijagramu sekvenci. Dostupni su različiti predefinisani tipovi fragmenata koji omogućavaju da se navedu drugačiji ishodi, paralelne poruke ili petlje.

U sledećem primeru, korisnik šalje zahtev ka stranici sa računima. Dva različita odgovora i uslova od kojih oni zavise su sadržana u fragmentu interakcije.



Slika broj 7. Fragment interakcije

Osobine fragmenta interakcije se mogu menjati iz njegove stranice sa osobinama. Da bi se otvorila stranica sa osobinama za fragment interakcije, potrebno je dvokliknuti levim tasterom miša na gornji levi deo simbola blizu taga operator. Najčešće korišćene osobine iz taba General su:

- Operator - određuje tip fragmenta. Može se birati između:
    - Alternative (alt) - fragment je podeljen na dva ili više međusobno isključivih regiona, pri čemu svaki od njih ima pridružen uslov. Poruka iz samo jednog od ovih regiona će biti izvršena u vreme izvršavanja programa.
    - Assertion (assert) - interakcija se mora desiti tačno onako kako je navedeno ili će biti neispravna
    - Break (break) - ako je pridruženi uslov tačan, prethodna interakcija se završava na kraju fragmenta
    - Consider (consider) - samo prikazane poruke su važne
    - Critical Region (critical) - ni jedna spoljašnja poruka ne može učestvovati u interakciji dok se date poruke ne izvrše
    - Ignore (ingore) - određene beznačajne poruke nisu prikazane
    - Loop (loop) - fragment interakcije će biti ponovljen određen broj puta
    - Negative (neg) - interakcija je nevalidna i ne može se desiti
    - Option (opt) - interakcija se dešava samo ako je uslov ispunjen
    - Parallel (par) - fragment je podeljen u dva ili više regiona, pri čemu će svi biti izvršeni u paraleli u vreme izvršavanja programa
    - Strict Sequencing (strict) - redosled poruka je strogo zadat
    - Weak Sequencing (seq) - redosled poruka je zadat na svakoj liniji života ali ne i između linija života
- Stereotip - proširuje značenje objekta izvan UML definicija
- Uslov - navodi bilo koji uslov koji je pridružen fragmentu. Ovo može biti provera varijable (npr. X > 3) ili u fragmentu petlje navođenje minimalne i maksimalne vrednosti koje predstavljaju broj izvršavanja (npr. 1,10). Za operatore *Consider* i *Ignore* ovo polje prikazuje listu pridruženih poruka. Ovo polje nije dostupno ukoliko fragment ne podržava uslove.

Fragment interakcije se može pomerati, može se menjati njegova veličina, itd. Kada simbol preklopi liniju života objekta, on se povezuje sa objektom automatski, a ova povezanost je prikazana sa malim ispupčenjem na gornjoj strani simbola gde se on susreće sa linijom života. Ukoliko pomerimo simbol od linije života objekta, on automatski prestaje da bude povezan.

2.6. Poruka

Poruka predstavlja komunikaciju između objekata. Objekti mogu sarađivati korišćenjem više tipova zahteva (pošalji signal, pozovi operaciju, kreiraj objekat, obriši postojeći objekat, itd.). Svi ovi zahtevi predstavljaju poruke.

Poruka ima objekat ili učesnika koji je šalje, objekat ili učesnika koji je prima i akciju. Akcija se izvršava na objektu ili učesniku koji primaju poruku. Mogu se kreirati i rekurzivne poruke pri čemu isti objekta i šalje i prima poruku.

Simbol za poruku je strelica koja pokazuje smer poruke. Pored ovoga poruka može sadržati:

- broj koji ukazuje redosled u kom će se poruke izvršavati
- ime poruke
- uslov
- povratnu vrednost
- argument

U dijagramima sekvenci poruka je prikazana kao horizontalna puna linija koja ide od linije života jednog objekta ili učesnika do linije života drugog objekta ili učesnika. Linija na sebi ima ime poruke. U dijagramima sekvenci može se birati između sledećih tipova poruka:

- poruka
- rekurzivna poruka
- poruka poziva procedure
- samopozivajuća poruka
- povratna poruka
- rekurzivna povratna poruka

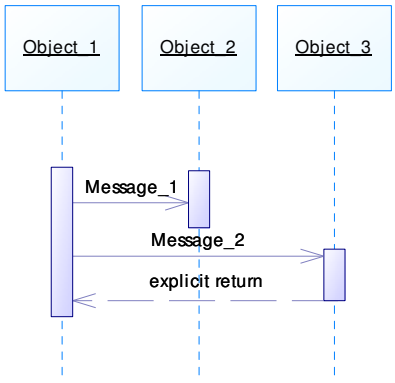
Osobine poruke se mogu menjati iz njene stranice sa osobinama. Da bi se otvorila ova stranice, potrebno je da se dvoklikne levim tasterom miša na simbol poruke. Najčešće korišćene osobine u karticama su:

- General
    - Name - navodi ime stavke, koje treba da bude jasno i da pokriva značenje stavke
    - Code - navodi tehničko ime objekta koje se koristi za generisanje koda ili skripti
    - Comment - opisni komentar poruke
    - Sequence number - omogućava da se manuelno dodaju brojevi redosleda poruka. Uglavnom se koristi u komunikacionim dijagramima da bi opisao redosled poruka, ali se može koristiti i u dijagramima sekvenci
    - Stereotype - proširuje značenje poruke izvan UML definicija
    - Sender - učesnik ili objekat iz kog kreće poruka. Pomoću alata koji su dostupni sa desne strane ove stavke moguće je kreiranje novih objekata, pregled stabla dostupnih objekata ili osobine trenutno odabranog objekta.
    - Receiver - učesnik ili objekat u kom se poruka završava. Pomoću alata koji su dostupni sa desne strane ove stavke moguće je kreiranje novih objekata, pregled stabla dostupnih objekata ili osobine trenutno odabranog objekta. Pored ovoga nalazi se i dugme Reverse direction pomoću kog se mogu zameniti Sender i Receiver. Smer poruke se ne može zameniti ako je to poruka koja kreira ili uništava objekat.
- Detail
    - Action - navodi tipove akcija poruke. Može se birati između:
        - Create - objekat koji šalje poruku instancira i inicijalizuje objekat koji prima poruku. Poruka sa akcijom kreiranja je prva poruka između pošaljioca i primaoca.
        - Destroy - objekat koji šalje poruku uništava objekat koji prima poruku. Oznaka X se prikazuje na liniji života objekta koji prima poruku. Poruka sa akcijom uništenja je poslednja poruka između pošaljioca i primaoca.
        - Self-Destroy - dostupna je samo ako je osobina toka kontrole podešena na *Return*. Objekat koji šalje poruku upozorava objekat koji prima poruku da će da uništi sebe. Oznaka X se prikazuje na liniji života objekta koji prima poruku. Poruka sa akcijom samouništenja je poslednja poruka između pošaljioca i primaoca.
    - Control flow - navodi način na koji se poruka šalje. Može se birati između:
        - Asynchronous - objekat koji šalje poruku ne čeka rezultat, već radi nešto drugo u međuvremenu.

        - Procedure Call - poziv procedure. Sekvenca se završava pre početka sledeće sekvence. Objekat koji šalje mora čekati odgovor ili kraj aktivacije.
        - Return - uglavnom se povezuje sa pozivom procedure. Može biti izostavljena jer se podrazumeva na kraju aktivacije.
        - Undefined - nije definisan tok kontrole.
    - Operation - povezuje poruku sa operacijom iz klase. Ukoliko je primalac poruke objekat, i objekat ima klasu, poruka, kao dinamički tok informacija, poziva operaciju. Na osnovu ovoga, moguće je povezati poruku sa postojećom operacijom klase, ali takođe i operacijama definisanim u klasama roditeljima, ili se može kreirati operacija iz liste operacije u stranici sa osobinama poruke. Ukoliko je operacija povezana sa porukom, moguće je zameniti ime poruke sa imenom metode koju objekat proziva.
    - Arguments - argumenti operacija
    - Return value - povratna vrednost funkcije
    - Predecessor list - lista koja govori koje poruke se moraju izvršiti pre trenutne poruke. Npr. poruke 1,2,4 se moraju izvršiti pre poruke 3 se zapisuje "1,2,4/3"
    - Condition - uslov povezan sa porukom
    - Begin time - vreme definisano od strane korisnika koje služi da se zadaju ograničenja. Npr. vreme početka = t1, vreme završetka = t2, ograničenje = {t2 - t1 < 30 sec}
    - End time - vreme definisano od strane korisnika koje služi da se zadaju ograničenja.
    - Support delay - navodi da poruka može imati trajanje. Simbol poruke može ići na dole. Ukoliko ova opcija nije odabrana, slanje poruke je trenutno a simbol je horizontalan. Ova opcija nije dostupna kod rekurzivnih poruka.

2.7. Tok kontrole

U podrazumevanom slučaju, poruka ima nedefinisan tok kontrole. Ukoliko želimo da učinimo dijagram čitljivijim, možemo dodati povratnu strelicu kako bi pokazali tačno vreme kada se akcija vraća pošaljiocu.



Slila broj 8. Primer Toka kontrole

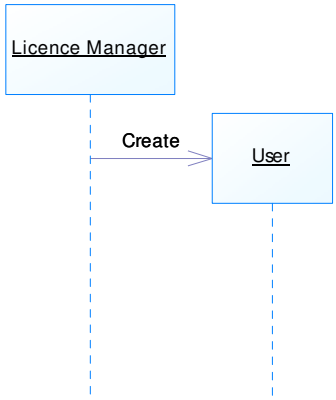
U zavisnosti od toka kontrole menja se i simbol kao što je prikazano u sledećoj tabeli:

Tok kontrole	Simbol
Asynchronous	
Procedure Call	
Return	
Undefined	

Slika broj 9. Prmeri vrste tokova kontrole

2.8. Poruka za kreiranje objekta

Poruka može kreirati objekat ako je to prva poruka koju objekat prima i ako osobinu poruke *Action* postavimo na *Create*. Ne može se kreirati učesnik niti se koristiti kreiranje kod rekurzivnih poruka.



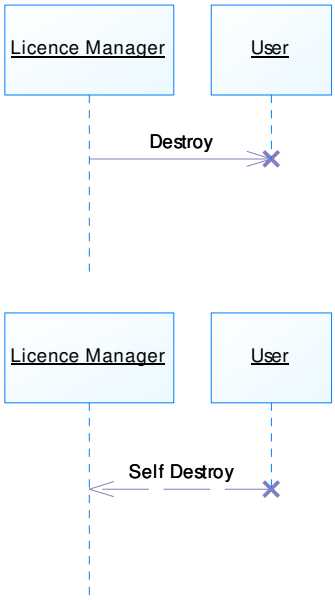
Slika broj 10. Primer poruke za kreiranje objekta

2.9. Poruka za uništavanje objekta

Poruka može uništiti objekat ukoliko je to poslednja poruka koju je objekat primio i ako osobinu *Action* postavimo na *Destroy*. Ne može se uništiti učesnik niti se koristiti uništavanje kod rekurzivnih poruka. Linija života objekta koji je uništen je označena sa X u tački preseka linije života i poruke.

Postoje dve forme poruke uništavanja:

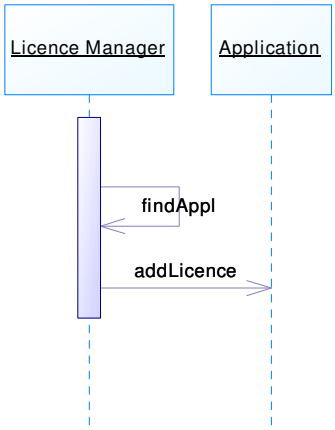
- poruka uništavanja
- poruka samouništavanja



Slika broj 11. Primer poruke uništavanja i poruke samouništavanja

2.10. Kreiranje rekurzivnih poruka

Poruka je rekurzivna kada objekat sam sebi šalje poruku. U ovom slučaju strelica počinje i završava se na liniji života istog objekta.

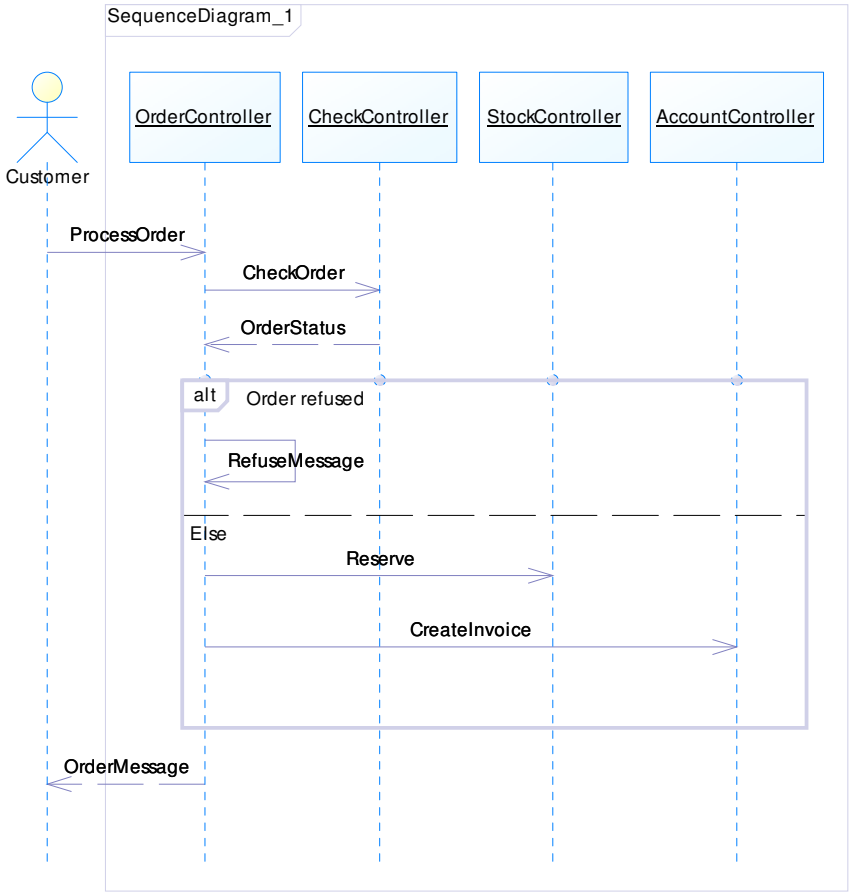


Slika broj 12. Primer rekurzivnog poziva

2.11. Poruke i kapije

U UML-u 2, može se slati poruka ka i od frejma interakcije koji okružuje dijagram sekvenci. Frejm predstavlja spoljašnju ivicu sistema (ili dela sistema) koji se modeluje i može služiti kao zamena za učesnika. Za poruku koja polazi iz tačke na frejmu se kaže da je poslata sa **ulazne kapije**, dok se za poruku koja stiže do frejma kaže da je stigla iz **spoljašnje kapije**.

Primer: Naručivanje robe:



Slika broj 13. Primer kapija

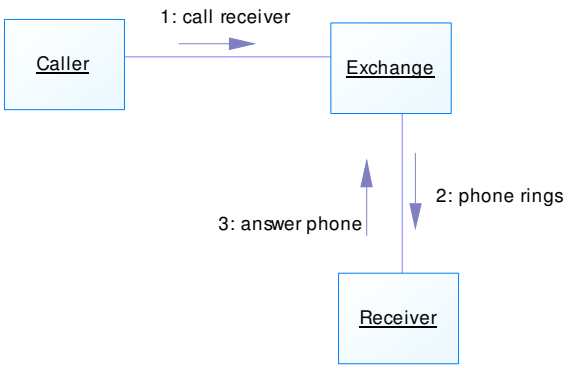
3. Komunikacioni dijagram

Komunikacioni dijagram je UML dijagram interakcije koji opisuje određenu funkcionalnost sistema tako što prikazuje skup hronoloških interakcija između objekata. On se može koristiti da ilustruje moguće načine upotrebe, izvršenje operacija ili način interakcije između klasa u sistemu.

Može se koristiti jedan ili više komunikacionih dijagrama da bi se prikazale sve mogućnosti složenih sistema.

Komunikacioni dijagram prikazuje isti tip informacija kao i sekvencijalni dijagram, sa razlikom da se koncentriše na strukturu objekata umesto na redosled poruka koje se između njih razmenjuju.

Komunikacioni dijagram prikazuje učesnike, objekte, njihove komunikacione linkove (koji se nazivaju i linkovi instanci) i poruke koje se razmenjuju između njih. Redosled u kom se poruke razmenjuju se prikazuje pomoću sekvence brojeva.



Slika broj 14. Primer komunikacionog dijagrama

3.1. Analiza slučajeva korišćenja

Komunikacioni dijagrami se mogu koristiti da bolje pojasne slučajeve korišćenja ili ponašanje sistema. Ovaj pristup je koristan kod analize zahteva jer može pomoći da se identifikuju klase i asocijacije koje nisu uočene na početku.

Često je potrebno kreirati više dijagrama da bi se opisali svi slučajevi korišćenja. U ovakvim situacijama, može biti korisno da se kreira komunikacioni dijagram da bi se otkrili svi značajni objekti pre identifikovanja klasa koje ih instanciraju.

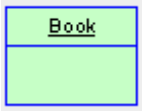
3.2. Analiza dijagrama klasa

Kreiranje komunikacionog dijagrama može biti prilika da se testira statički model na nivou koncepta. On može predstavljati scenario u kom su klase iz dijagrama klasa instancirane da bi se kreirali objekti koji su neophodni da bi se scenario pokrenuo.

On dopunjuje dijagram klasa koji predstavlja statičku strukturu sistema jer navodi ponašanje klasa, interfejsa i moguću upotrebu njihovih operacija.

4. Objekti u komunikacionom dijagramu

U komunikacionom dijagramu se kreiraju sledeći objekti:

Objekat	Alat	Simbol	Opis
Učesnik			Spoljašnja osoba, proces ili nešto što vrši interakciju sa sistemom, podsistemom ili klasom.
Objekat			Instanca klase.
Link instance			Komunikacioni link između dva objekta
Poruka			Interakcija koja prenosi informacije

Slika broj 15. Objekti komunikacionog dijagrama

4.1. Učesnik

Učesnik je spoljašnji korisnik ili grupa korisnika koji vrše interakciju sa sistemom. Učesnici mogu biti ljudi ili drugi spoljašnji sistemi. Učesnici su uglavnom oni entiteti čije ponašanje ne možemo da kontrolišemo ili menjamo, jer oni nisu deo sistema koji opisujemo.

U komunikacionim dijagramima učesnik može biti povezan sa objektom preko linka instance i može slati ili primati poruke.

4.2. Objekat

Objekat se definiše kao deo sistema koji se opisuje. Mogu se predstaviti tri moguće situacije vezane za prikaz objekta:

- objekat nije instanca klase - on ima samo ime
- objekat predstavlja instancu klase - ima ime i klasu
- objekat predstavlja instancu klase ali zapravo predstavlja bilo koju instancu klase - ima klasu ali nema ime.

4.3. Link instance

Link instance predstavlja konekciju između dva objekta. On se prikazuje kao puna linija između dva objekta. Može biti prikazan između:

- dva objekta
- objekta i učesnika

Simbol za link instance može sadržati i više poruka koje su povezane sa njim. Link instance sadrži redosled poruka koje su pridružene za njega.

4.4. Poruka

Poruka predstavlja komunikaciju između objekata. Objekti mogu sarađivati korišćenjem više tipova zahteva (pošalji signal, pozovi operaciju, kreiraj objekat, obriši postojeći objekat, itd.). Svi ovi zahtevi predstavljaju poruke.

Poruka ima objekat ili učesnika koji je šalje, objekat ili učesnika koji je prima i akciju. Akcija se izvršava na objektu ili učesniku koji primaju poruku. Mogu se kreirati i rekurzivne poruke pri čemu isti objekta i šalje i prima poruku.

Simbol za poruku je strelica koja pokazuje smer poruke.

U komunikacionom dijagramu svaka poruka je povezana sa linkom instance. Link instance može imati više pridruženih poruka, ali svaka poruka može biti pridružena samo jednom linku instance. Uništavanje linka instance uništava sve poruke pridružene njemu.

Za vežbu na času / Domaći zadatak

Kreirati sekvencijalni i komunikacioni dijagram za učenika koji koristi sistem školske biblioteke. Za dati sistem potrebno je kreirati dijagrame za iznajmljivanje knjiga i dijagrame za vraćanje knjiga. Sistem treba da pruži sledeće funkcionalnosti:

- upravljanje korisnicima
  - prijavljivanje na sistem
  - odjava sa sistema
- upravljanje knjigama
  - pretraga knjiga
  - upis u listu čekanja
- usluge iznajmljivanja knjiga
  - iznajmljivanje knjige
  - vraćanje knjige
  - računanje vremena do kada knjiga treba da bude vraćena
- formiranje izveštaja
  - pregled knjiga koje je učenik pročitao
  - izveštaj o nevraćenim knjigama

Sistem radi na sledeći način:

Učenik se prvo prijavljuje na sistem i zahteva knjigu koja mu je potrebna. Nakon ovoga se vrši pretraga dostupnih knjiga u biblioteci. Ukoliko je knjiga zauzeta, on se upisuje u listu čekanja. Ako je knjiga nije zauzeta, učenik iznajmljuje knjigu i dobija informaciju do kada mora da je vrati. On tada dobija izveštaj u kome može videti knjige koje nije vratio.

Prilikom vraćanja knjige, učenik se prijavljuje na sistem i vraća knjigu. On tada dobija izveštaj u kome može videti knjige koje je pročitao.

Za rešavanje zadatka obavezno koristiti Reference interakcije i Fragmente interakcije.

Pitanja za obnavljanje gradiva:

1. Čemu služe dijagrami sekvence?
2. Čemu služi učesnik u dijagramima sekvence?
3. Čemu služi objekat?
4. Čemu služi aktivacija?

5. Čemu služi referenca interakcije?
6. Čemu služi fragment interakcije?
7. Čemu služi poruka i kakve mogu biti?
8. Čemu služi komunikacioni dijagram?
9. Koji su osnovni elementi u komunikacionim dijagramima?