Oblikovanje programske potpore

2012./2013. grupa P01

Primjena UML-a u inženjerstvu zahtjeva

Prof.dr.sc. Vlado Sruk

Sveučilište u Zagrebu
Fakultet elektrotehnike i računarstva
Zavod za elektroniku, mikroel., računalne i inteligentne sustave

Tema

- Primjena UML-a (engl. Unified Modeling Language) u inženjerstvu zahtjeva

- Primjena UML-a
- Razvoj i elementi jezika
- Obrasci uporabe
- Dijagram interakcija

Literatura

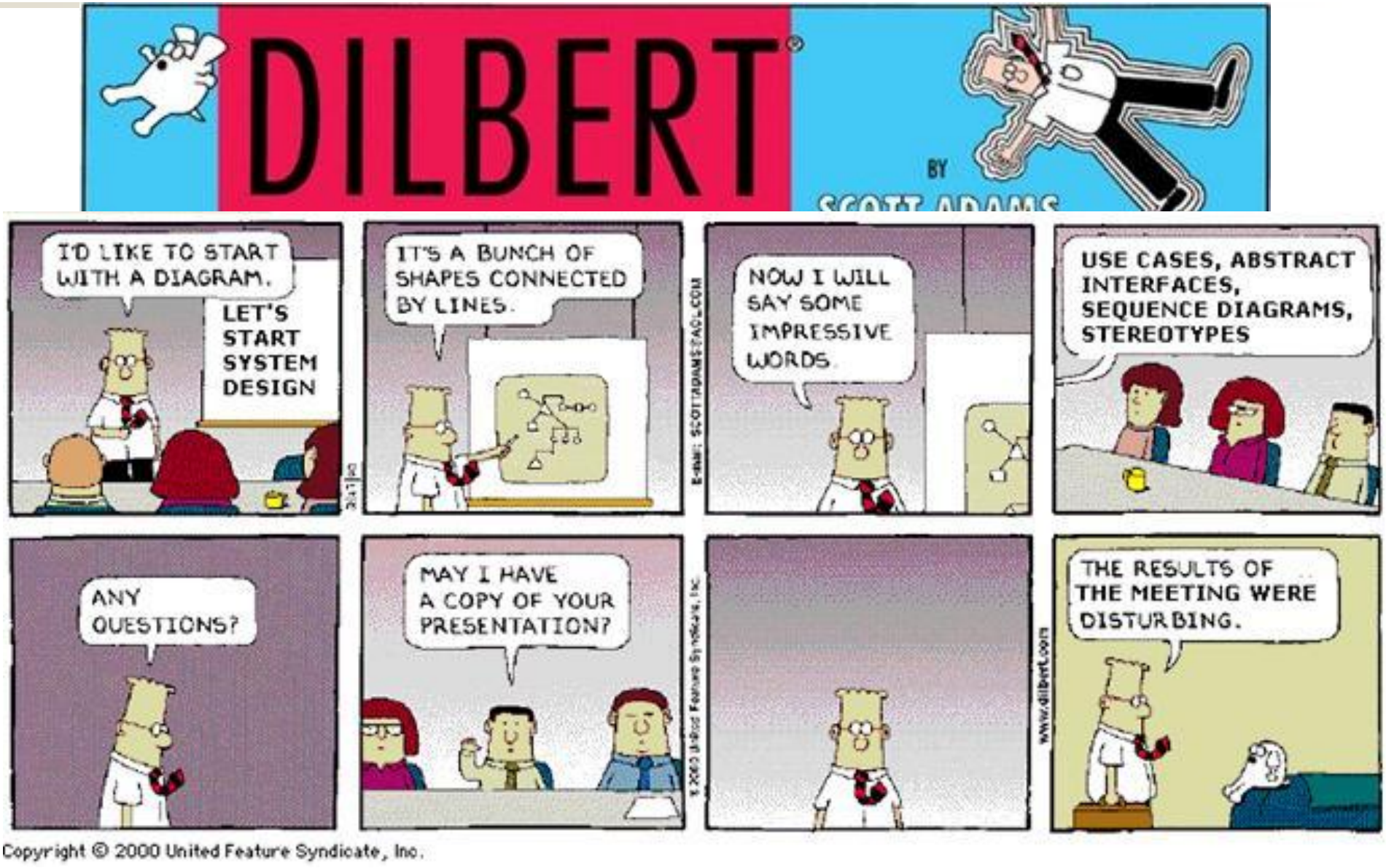
- Sommerville, I., ***Software engineering***, 8th ed., Addison-Wesley, 2007.
- Grady Booch, James Rumbaugh, Ivar Jacobson: ***Unified Modeling Language User Guide*****, 2nd Edition, 2005**
- **Simon Bennett, John Skelton, Ken Lunn: *Schaum's Outline of UML*, Second Edition, 2005**
- WWW
- Rational Software Architect
  - http://publib.boulder.ibm.com/infocenter/rtnlhelp/v6r0m0/index.jsp?topic=/com.ibm.xtools.modeler.doc/topics/cextend.html

Unified Modeling Language

- UML je jezik za:
  - vizualizaciju;
  - specifikaciju;
  - oblikovanje i
  - dokumentiranje

  Artefakata programske potpore.
- Omogućava različite poglede na model
- ‘de facto’ standardni jezik programskog oblikovanja
- UML je posebice prikladan za specificiranje ***objektno usmjerene*** arhitekture programske potpore.
- Dijelovi UML-a pogodni su u specificiranju i drugih arhitektura.
- Omogućava
  - Višestruke međusobno povezane poglede
  - Poluformalnu semantiku izraženu kao metamodel
  - Jezik za opis formalnih logičkih ograničenja

Percepcija UML-a



Povijest

- UML: Unified Modeling Language
  - Rumbaugh et al.: OMT 1991
    - Baze podataka i Entity Relation model
  - Jacobson: OOSE 1992
    - use cases / requirements
  - Booch: Booch notation 1994
    - Oblikovanje jezika, strukturni aspekt i nasljeđivanje

- “The Three Amigos” 1997
  - unified really means ”joint effort makes more money compared to notational wars”

Potreba

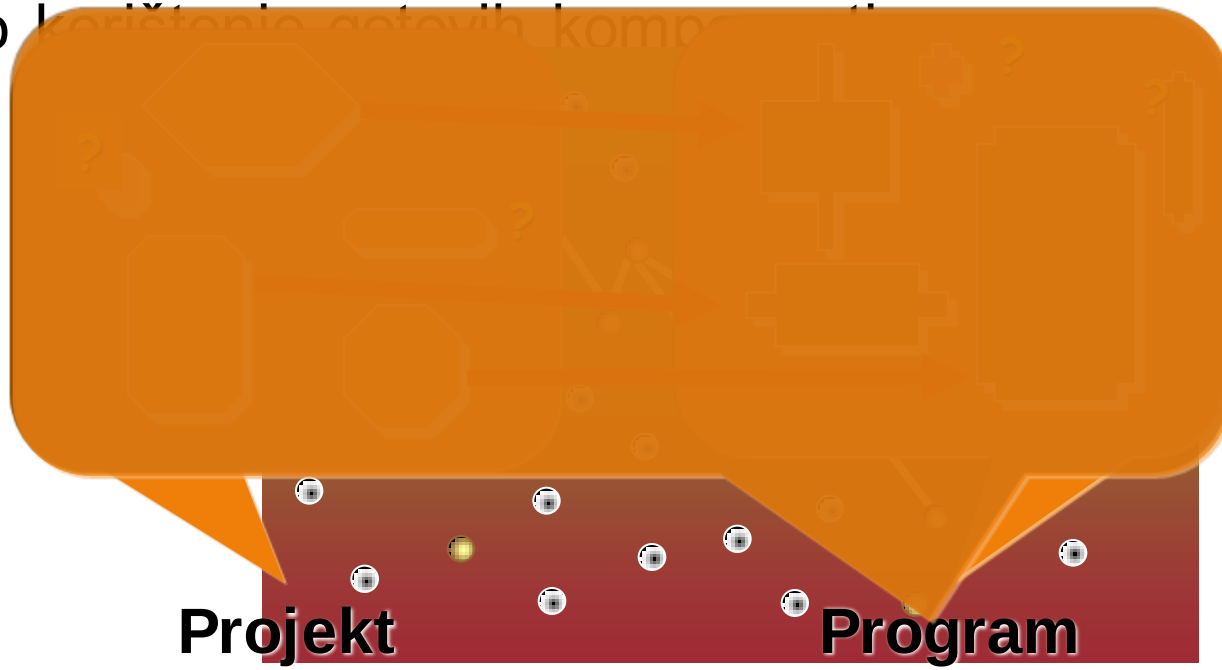
- Osmisliti jezik koji osigurava jednostavan riječnik i pravila kombiniranja riječi u svrhu komunikacije.
- U jeziku modeliranja riječnik i pravila su usmjerena na konceptualnu i fizičku reprezentaciju sustava.

$\Rightarrow$ UML standard

- ”A language provides a ***vocabulary*** and ***the rules for combining words*** […] for the purpose of ***communication***. A *modeling* language is a language whose vocabulary and rules focus on the conceptual and physical representation of a system. A modeling language such as the UML is thus a standard language for ***software blueprints***.”
  - *From ”UML user guide”*

UML: osnovna svojstva

- Osnovne ideje
  - obuhvatiti i opisati poslovne procese
  - poboljšati komunikaciju
  - pomoć u borbi s kompleksnošću
  - definirati logičku arhitekturu sustava
  - omogućiti ponovno korištenje gotovih komponenti
- Namjena:
  - vizualizacija
  - specificiranje
  - konstrukcija
  - dokumentiranje

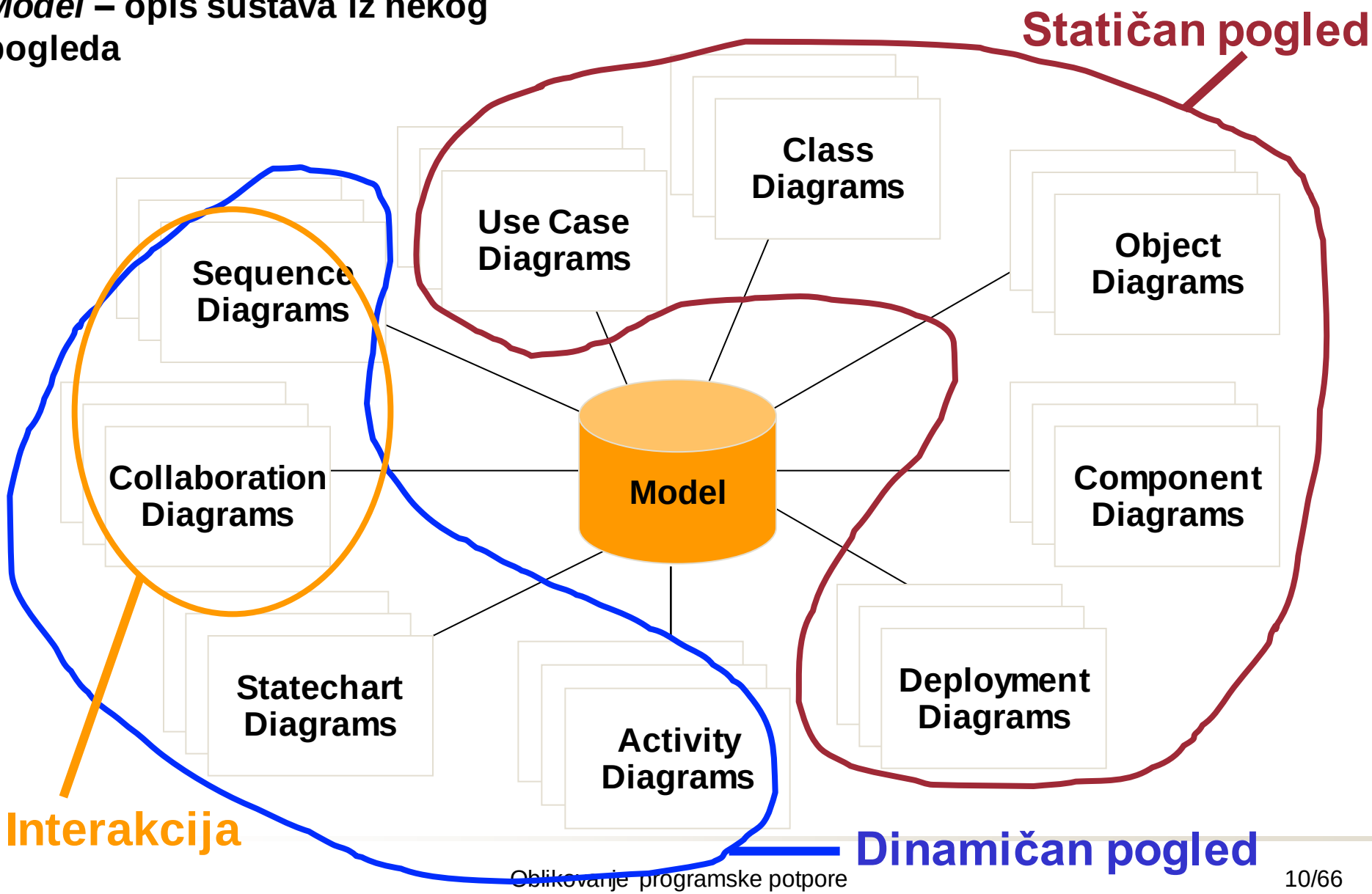


Osnovni elementi UML-a

- Stvari (*engl. things*)
  - strukturne stvari (*engl. structural things*)
    - *klasa; sučelje; suradnja; obrasci uporabe; aktivna klasa; komponenta; čvor*
    - aktori, signali, procesi i niti, aplikacije, dokumenti, datoteke, biblioteke, stranice, tablice
  - stvari ponašanja (*engl. behavioral things*)
    - interakcija; stanje
  - stvari grupiranja (*engl. grouping things*)
    - Organizacija elemenata u grupe (samo konceptualno, hijerarhija), paketi
  - stvari označavanja (*engl. annotation things*)
    - opisni elementi, formalni/neformalni opis
- Relacije (*engl. relationships*)
  - Asocijacije *(eng. association)*
  - Generalizacije *(eng. generalization)*
  - Realizacije *(eng. realization)*
  - Ovisnosti *(eng. dependency)*
- Dijagrami (*engl. diagrams*)
  - Dijagram klasa; objekata; slučajeva korištenja; toka; suradnji; stanja; aktivnosti; komponenata;....

Modeli, Pogledi, Dijagrami

Model **– opis sustava iz nekog pogleda**

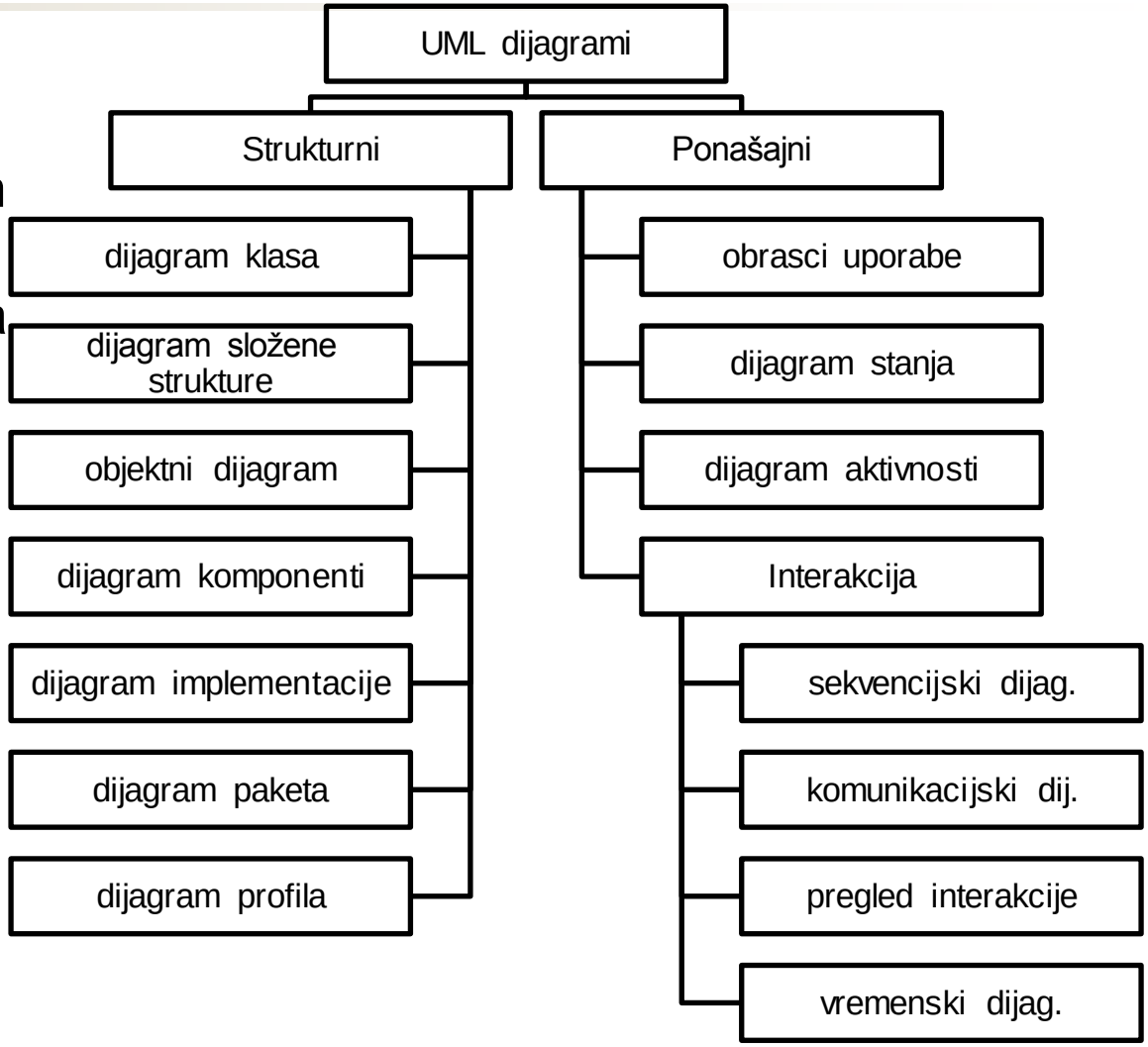


Dijagrami

- Pogled na model
  - Iz perspektive dionika
  - Djelomična reprezentacija sustava
  - Semantički konzistentan s ostalim pogledima
- Standardni dijagrami (UML 1.1)
  - **statični:** dijagram obrazaca uporabe, dijagram razreda, dijagram objekata, dijagram komponenata, dijagram razmještaja
    - ***use case, class, object*, component, deployment**
  - **dinamični:** sekvencijski dijagram, komunikacijski (kolaboracijski) dijagram, dijagram stanja ("statechart"), dijagram aktivnosti
    - ***sequence*, collaboration, statechart, activity**
  - uz manje promjene najčešće upotrebljavani dijagrami i u novijim inačicama UML-a
- Specifičnost dijagrama
  - Svaki ima vlastitu sintaksu i semantiku ☹ ?
- Evolucija UML-a
  - Veljača 2009. UML 2.2
  - 0žujak 2011. UML 2.4- beta
  - - razrada 2.5

UML 2.4

- Evoluiraju tijekom procesa oblikovanja, te se mijenjaju donošenjem odluka o oblikovanju i proširuju novim detaljima



UML 2.0

- **1. Structural Modeling Diagrams**
  Structure diagramdefine the static architecture of a model. They are used to model the 'things' that make up a model - the classes, objects, interfaces and physical components. In addition, they are used to model the relationships and dependencies between elements.
- Package diagramare used to divide the model into logical containers, or 'packages', and describe the interactions between them at a high level.
- Class or Structural diagram define the basic building blocks of a model: the types, classes and general materials used to construct a full model.
- Object diagramshow how instances of structural elements are related and used at run-time.
- Composite Structure diagramprovide a means of layering an element's structure and focusing on inner detail, construction and relationships.
- Component diagramare used to model higher level or more complex structures, usually built up from one or more classes, and providing a well defined interface.
- Deployment diagramshow the physical disposition of significant artifacts within a real-world setting.

2. Behavioral Modeling Diagrams
Behavior diagramcapture the varieties of interaction and instantaneous states within a model as it 'executes' over time; tracking how the system will act in a real-world environment, and observing the effects of an operation or event, including its results.

- Use Case diagram are used to model user/system interactions. They define behavior, requirements and constraints in the form of scripts or scenarios.
- Activity diagram have a wide number of uses, from defining basic program flow, to capturing the decision points and actions within any generalized process.
- State Machine diagramare essential to understanding the instant to instant condition, or "run state" of a model when it executes.
- Communication diagramshow the network, and sequence, of messages or communications between objects at run-time, during a collaboration instance.
- Sequence diagram are closely related to communication diagramand show the sequence of messages passed between objects using a vertical timeline.
- Timing diagramfuse sequence and state diagramto provide a view of an object's state over time, and messages which modify that state.
- Interaction Overview diagramfuse activity and sequence diagramto allow interaction fragments to be easily combined with decision points and flows.

UML dijagrami

Obrasci uporabe	*engl. use case diagrams*
Sekvencijski dijagram	*engl. sequence diagrams*
Komunikacijski dijagram	*engl. communication diagrams*
Dijagram stanja	*engl. state machine diagrams*
Dijagram aktivnosti	*engl. activity diagrams*
Dijagram komponentni	*engl. component diagrams*
Dijagram implementacije	*engl. deployment diagrams*
Dijagram paketa	*engl. package diagrams*
Dijagram pregleda interakcije	*engl. interaction overview diagrams*
Vremenski dijagram	*engl. timing diagrams*
Dijagram profila	*engl. profile diagram*
Dijagram klasa	*engl. class diagrams*
Dijagram objekata	*engl. object diagrams*
Dijagram složene strukture	*engl. composite structure diagrams*

Obrasci uporabe

- *engl. Use cases*
- Pogled na sustav koji naglašava njegovo vanjsko ponašanje prema korisniku
- Dijeli funkcionalnost sustava u skup obrazaca uporabe (transakcija) koji su značajni korisniku (aktoru)
- Skup obrazaca uporabe opisuje ***sve moguće interakcije sustava***.
- Uz obrasce uporabe, dodatno se mogu koristiti i dijagrami sekvenci kako bi se detaljno opisao tijek događaja.

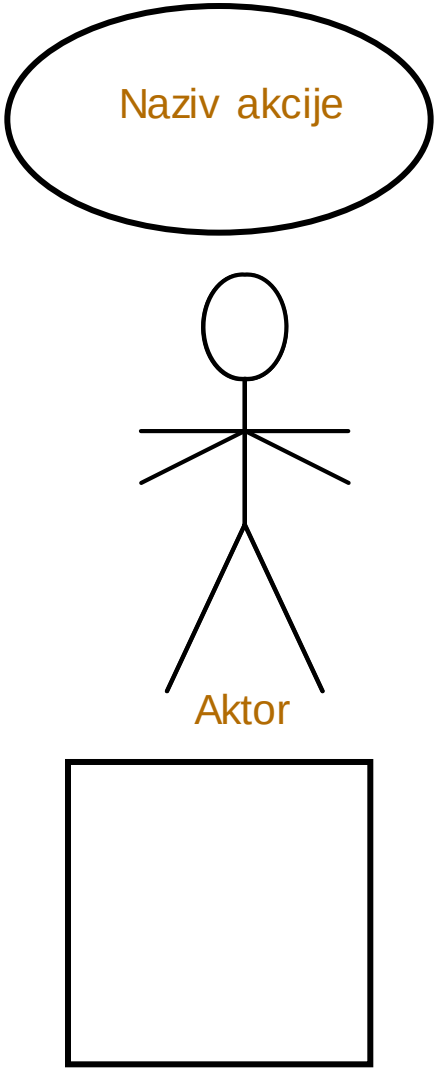
Dijagram obrazaca uporabe

- Opisuje obrasce uporabe, aktore i njihove odnose.
- Jednostavan grafički prikaz granica sustava, tko/što koristi sustav, te prikazuje kako ga koristi
- Detalji unutar dijagrama obrazaca uporabe mogu se predstaviti tekstom ili dodatnim dijagramima interakcije između elemenata sustava.

- Vrste obrazaca uporabe:
  - Dijagram – uobičajeno u uporabi
  - Tekstualni opis

Elementi obrazaca uporabe

- Osnovni (jezgreni) elementi:
- ***Obrazac uporabe*** *engl. use case*
  - Sekvenca akcija (uključujući varijante) koje sustav ili drugi entitet obavlja u interakciji s aktorima sustava (≡ funkcionalnost)
- ***Aktor*** *– engl. actor*
  - Vanjski objekt koji komunicira sa sustavom
  - Jedinstveno ime (+ po potrebi opis)
  - Koherentan skup uloga koje imaju korisnici u interakciji s obrascima uporabe.
  - Uloga korisnika u sustavu
  - Jedan korisnik – više aktora
  - Vanjski sustav
- ***Granica sustava*** *– engl. system boundary*
  - granica između fizikalnog sustava i različitih aktora koji su u interakciji s fizikalnim sustavom.



Definicije

- “A ***use case*** specifies the behavior of a system or a part of a system, and is a description of a set of sequences of actions, including variants, that a system performs to yield an observable result of value to an actor.”
  - *The UML User Guide, [Booch,99]*
- “An ***actor*** is an idealization of an ***external person, process, or thing interacting*** with a system, subsystem, or class. An actor characterizes the interactions that outside users may have with the system.”
  - *- The UML Reference Manual, [Rumbaugh,99]*

Aktori

- Za imenovanje upotrijebiti imenice
- Potrebno opisati njihovu ulogu u interakciji sa sustavom
- Definirati opseg sustava, identificirati elemente na rubu sustava i elemente o kojima sustav ovisi

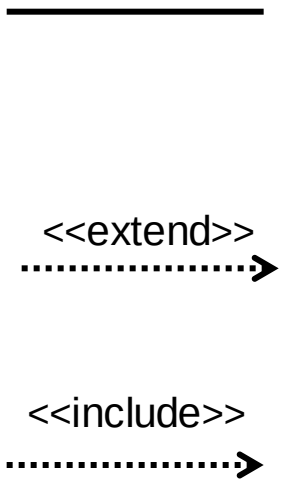
- Obrasci uporabe se pišu iz perspektive aktora!
  - Što radi koji aktor?
  - Za aktivost aktora definira se obrazac uporabe
    - Opišite ju tekstom
  - Zadržite razinu apstrakcije

Obrazac uporabe

- Imenovanje: glagol-imenica
- Opisati uvjete početka, završetka, tijek razmjene inf., nefukcijska svojstva, ... – tekstualno + drugi UML dijagrami
- Definira doseg sustava i funkcionalnost koju podržava u sustavu te elemente o kojima ovisi
- Potpomaže aktorima u svrhu ostvarivanja ciljeva
  - Obrasci uporabe predstavljaju funkcionalnosti sustava i odgovornosti (*engl. responsibilities*)

Osnovne poveznice obrazaca uporabe

- Poveznice – jezgreni odnosi
  - (*engl. Communication relationship*)
- ***Pridruživanje/asocijacija*** *– engl. association*
  - komunikacija instancije aktora i instancije obrasca uporabe
- ***Proširenje*** *– engl. extend*
  - Odnos od proširene uporabe do osnovne uporabe
- ***Obuhvaćanje/nužno sadrži*** *– engl. include*
  - Odnos od osnovnog obrasca do uključenog obrasca
  - Definira ponašanje uključenog obrasca u odnosu na osnovni obrazac uporabe.
  - Osnovni obrazac sadrži ponašanje definirano u drugom obrascu.
- ***Poopćenje*** *– engl. generalization*
  - odnos između općeg i specifičnog
- Opis višestrukih vrijednosti
  - Jednoznačan 1; 6; *
  - interval vrijednosti 0..1

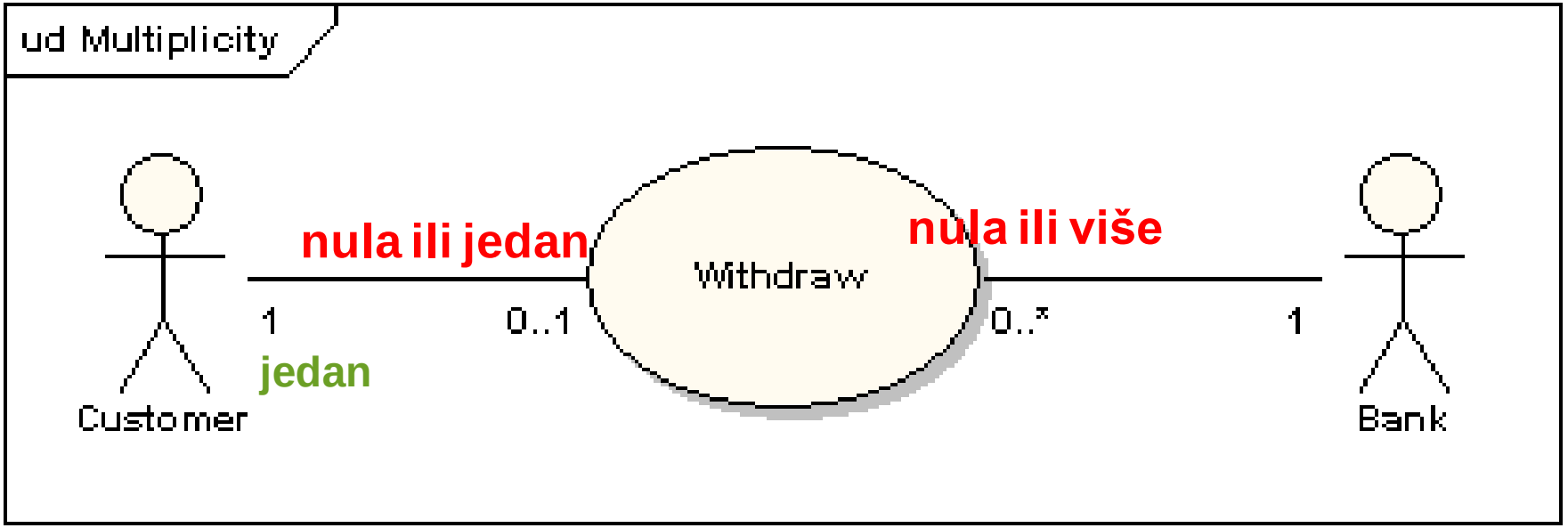


Elementi obrasca uporabe

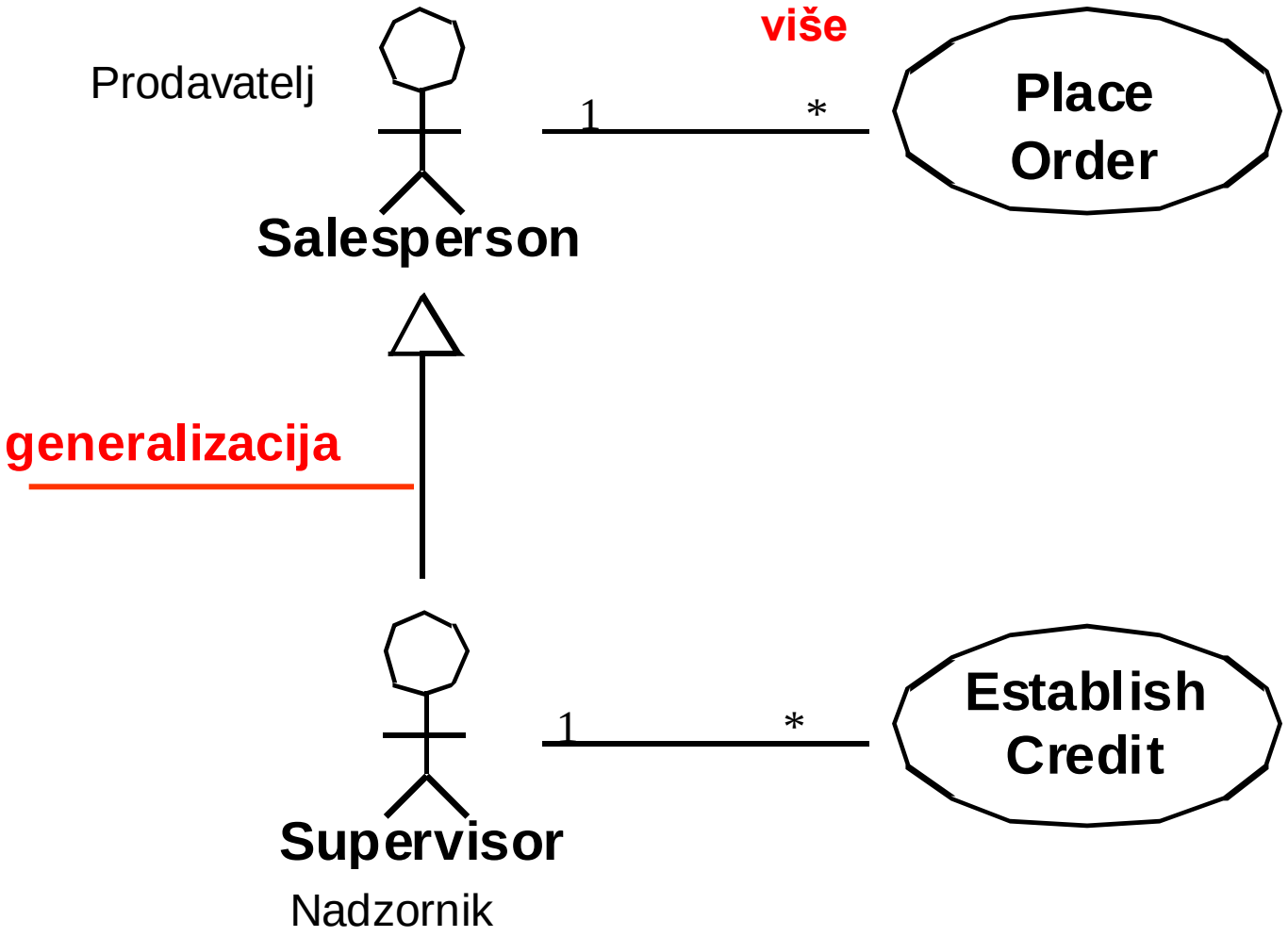
Element	Opis
Redni broj	Jedinstveni identifikacijski broj
Namjena	Koja je namjena sustava
Naziv	Jasno govoreće ime
Opis	Kratak opis
Glavni aktor	Tko je glavni aktor na kojeg se obrazac odnosi
Preduvjeti	Što mora biti zadovoljeno za aktivaciju obrasca
Pokretač	Događaj koji aktivira obrazac
Opis osnovnog tijeka	Opis idealnog tijeka događaja
Opis mogućih odstupanja	Najznačajnije alternative i iznimke

Višestrukost u obrascima uporabe

- Spojnica u obrascima uporabe može opcijski imati ***višestrukost*** (*engl. multiplicity values*) na svakom kraju.
  - Npr. klijent može imati najviše jednu isplatu u jednom času
  - banka može imati po volji broj transakcija (klijenata koji istovremeno obavljaju isplatu). (?)



Odnosi između aktora

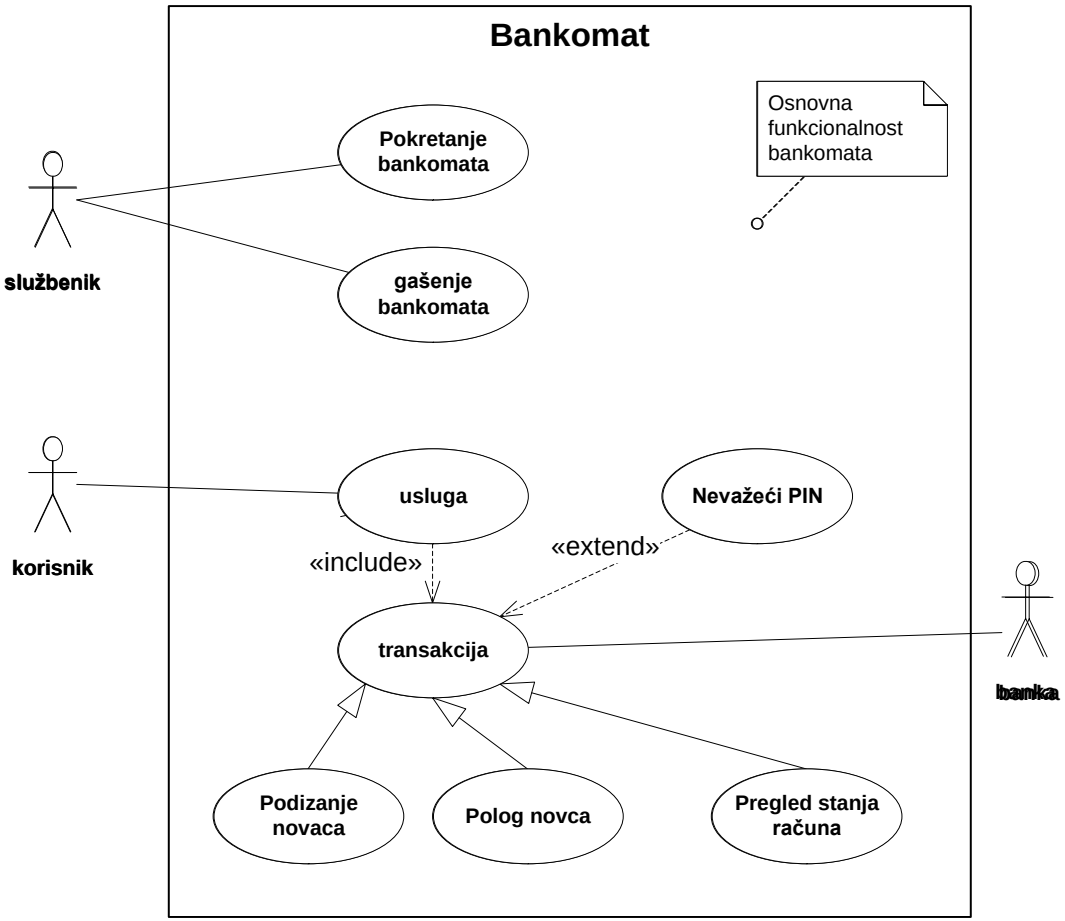


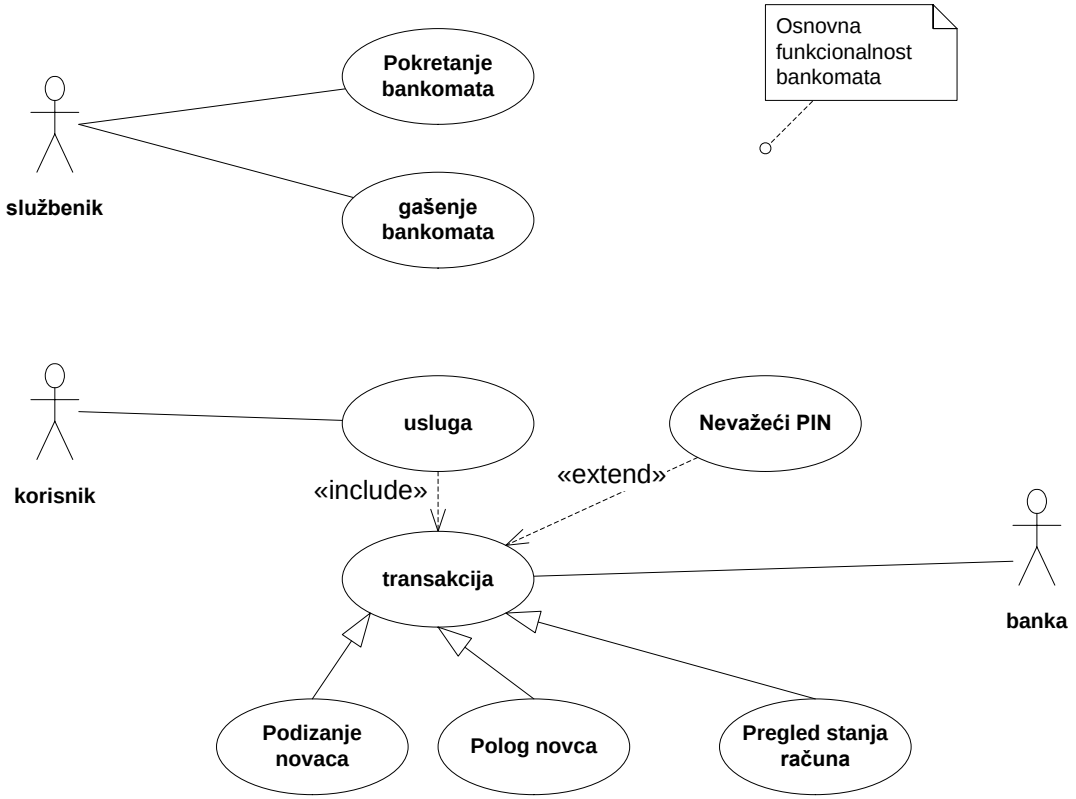
Tekstovni obrazac uporabe

- Primjer: Promjena rute leta

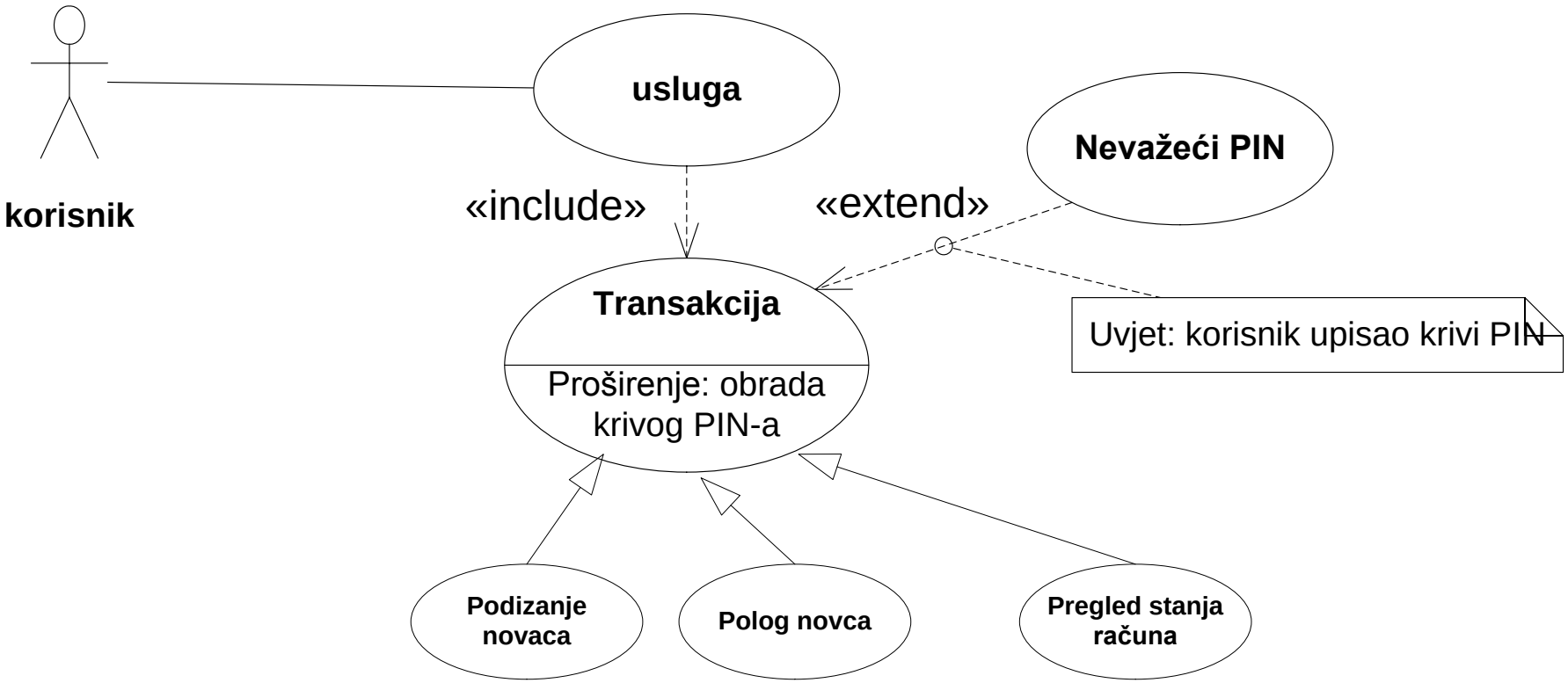
- **Aktori:**
  - putnik, baza računa klijenta (s planom puta), rezervacijski sustav avio kompanije.
- **Preduvjeti:**
  - Putnik se prijavio na sustav i odabrao opciju “promjena leta”.
- **Temeljni tijek transakcija**
  - Sustav dohvaća putnikov bankovni račun i plan puta iz baze.
  - Sustav pita putnika da odabere dio plana puta koji želi mijenjati; putnik selektira segment puta.
  - Sustav pita putnika za novi odlaznu i dolaznu destinaciju; putnik daje traženu informaciju.
  - Ako je let moguć, tada …
  - …
  - Sustav prikazuje sažetak transakcije..
- **Alternativni tijek transakcija**
  - Ako let nije moguć, tada …

Primjer: Dijagram obrazaca uporabe




Osnovna funkcionalnost bankomata
Pokretanje bankomata
gašenje bankomata
službenik
usluga
Nevažeći PIN
«extend»
«include»
korisnik
transakcija
banka
Podizanje novaca
Polog novca
Pregled stanja računa

Relacije u obrascima uporabe



Primjena modeliranja obrascima uporabe

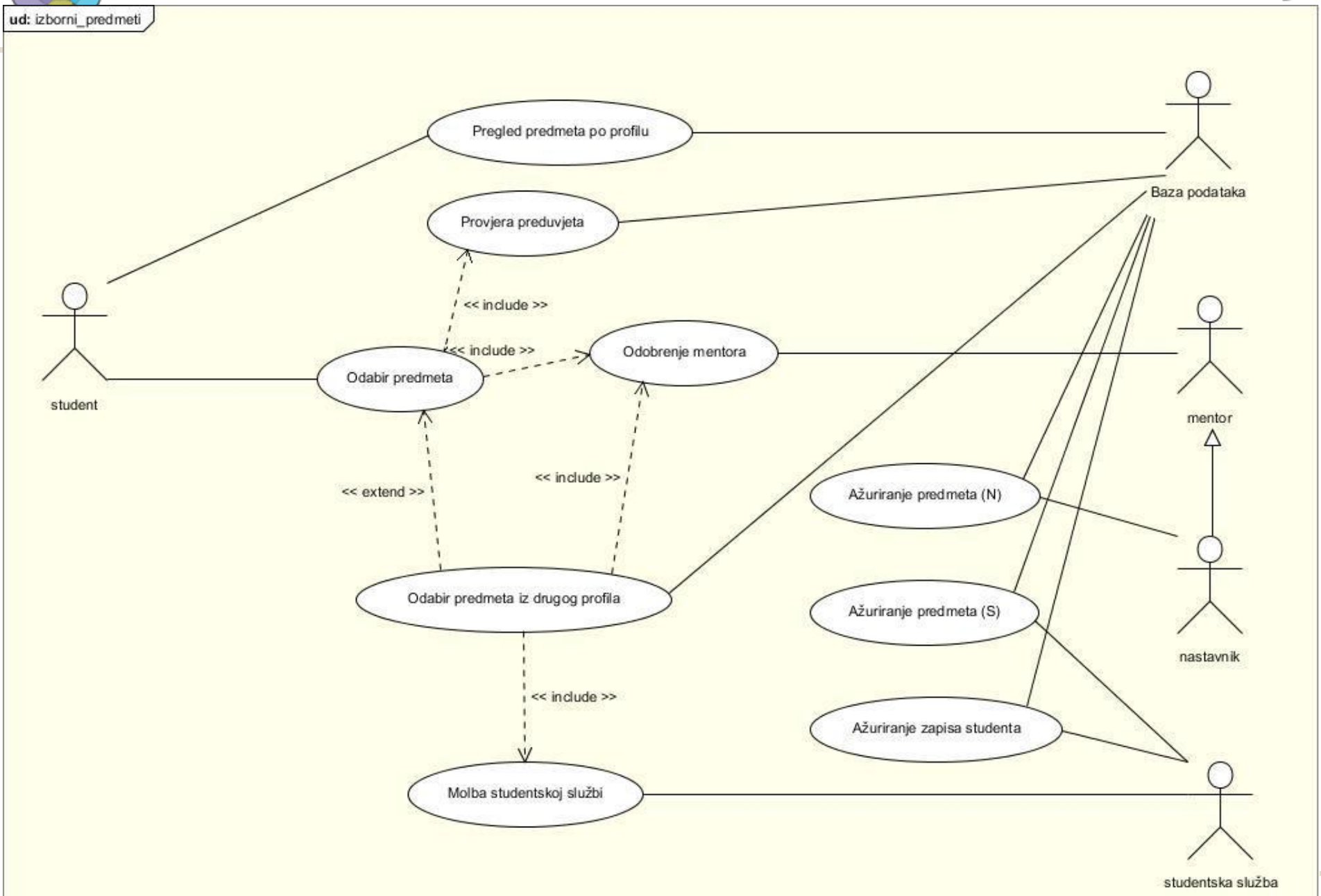
- ***Zahtjevi korisnika***
- ***Scenariji ispitivanja sustava*** (*engl. test scenarios*)

- Za primjenu metode oblikovanja programske potpore zasnovanu na obrascima uporabe
  - Započni s obrascima uporabe i iz njih izvedi strukturne i ponašajne (*engl. behavioral*) modele sustava.
- Ili ako ne koristiš metodu oblikovanja programske potpore zasnovanu na obrascima uporabe
  - Osiguraj da su obrasci uporabe konzistentni s strukturnim i ponašajnim modelima.

Preporuke u oblikovanju obrazaca uporabe

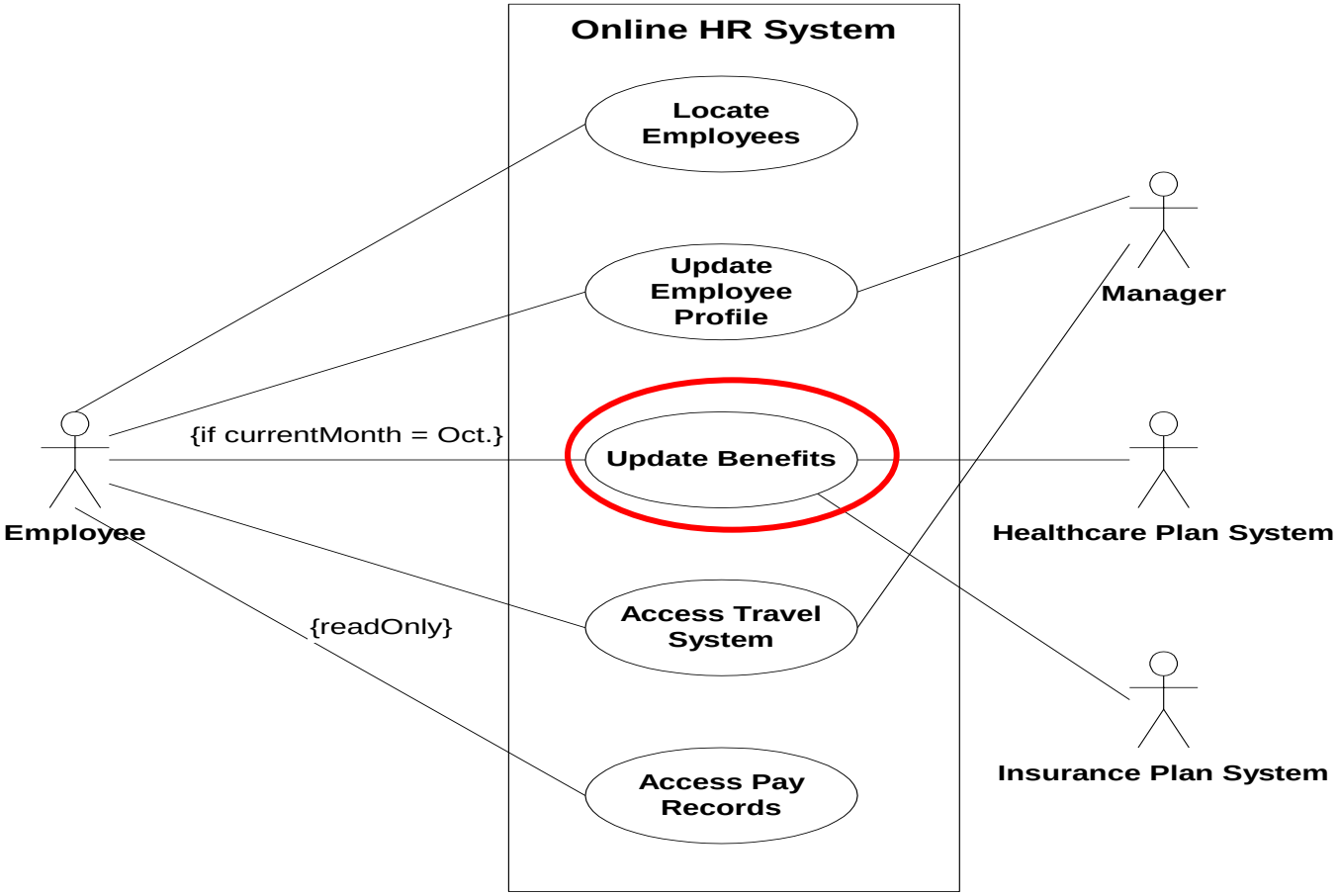
- Svaki obrazac uporabe mora sadržavati značajan dio uporabe sustava i biti razumljiv ekspertima domene i programerima.
- Ako se obrasci uporabe definiraju tekstom sve imenice i glagole treba koristiti razumljivo i konzistentno kako bi se kasnije mogli definirati ostali (UML) dijagrami.
- Ako su obrasci uporabe nužni, koristi <<include>>.
- Obrasci uporabe su kompletirani a postoje opcije, koristi <<extend>>.
- Dijagram obrazaca uporabe treba sadržavati obrasce uporabe jednake apstraktne razine.
- Uključiti samo zaista potrebne aktore.
- Veliki broj obrazaca uporabe treba organizirati u posebne odvojene dijagrame i pakete.

Primjer – odabir izbornih predmeta (hipotetski sustav)

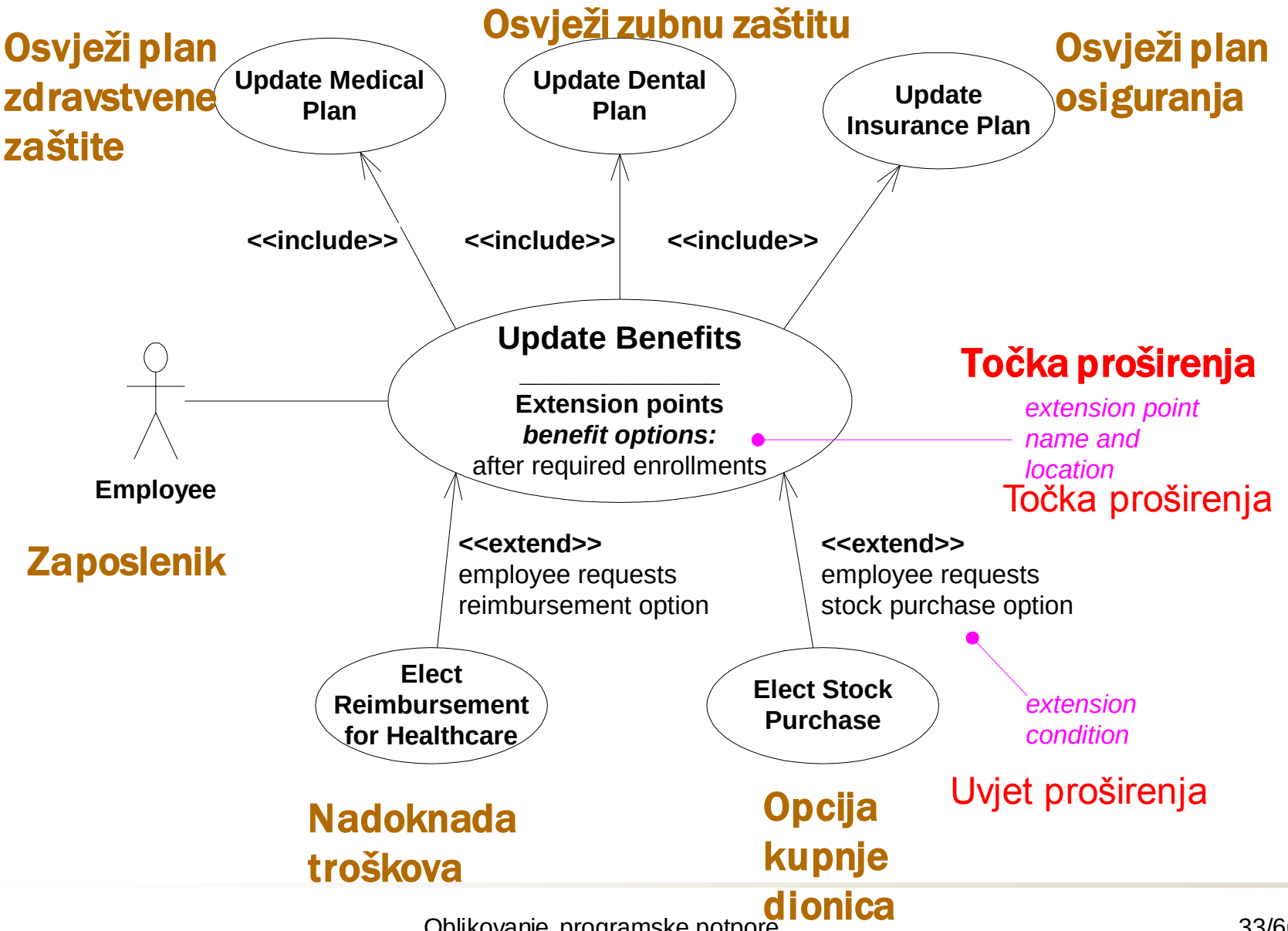


Primjer: Upravljanje ljudskim resursima

- "on-line" sustav za upravljanje ljudskim resursima



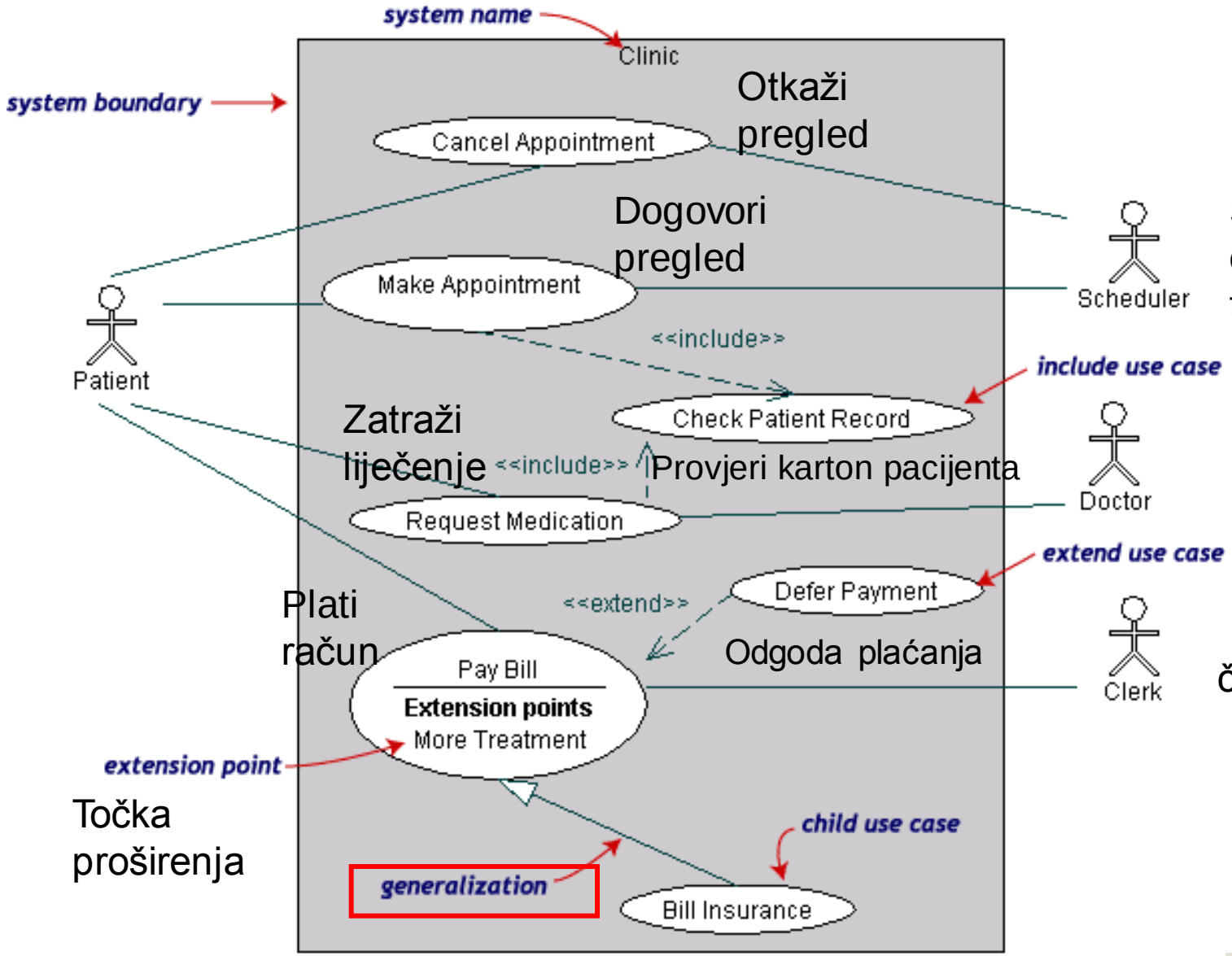
Primjer: Akcija - "osvježi pogodnosti"



Primjer tekstualnog opisa

- **Aktori:** zaposlenik, baza računa zaposlenika, sustav zdravstvene zaštite, sustav (zdravstvenog) osiguranja.
- **Preduvjeti:**
  - Zaposlenik se prijavio na sustav i odabrao opciju: '`update benefits`'.
- **Temeljni slijed:**
  - Sustav dohvaća zaposlenikov račun iz baze.
  - Sustav traži da zaposlenik odabere tip plana zdravstvene zaštite; **include** `Update Medical Plan`.
  - Sustav traži da zaposlenik odabere tip plana zubne zaštite; **include** `Update Dental Plan`.
  - …
- **Alternativni sljedovi:**
  - Ako tip plana zdravstvene zaštite kojeg je zaposlenik odabrao nije ponuđen u njegom mjestu stanovanja, sustav traži odabir drugog plana zaštite.

Primjer: Ordinacija



Obrasci uporabe: pravila

- Opisuje tipičnu uporabu sustava
- Mora specificirati najvažnije funkcionalne zahtjeve
- Detaljna specifikacija može dovesti do otkrivanja novih zahtjeva
- Opisuje značajke važne dizajneru i naručitelju
- “igrokaz”
- Kompletan opis početak, sadržaj i kraj
- Jednostavnost i izravnost
- Jednoznačnost
- Jedna stranica
- Specifikacija pogodna za razvoj i testiranje

Uporaba UML obrazaca uporabe

- UML dijagrami i tekstualni opisi obrazaca uporabe ***modeliraju funkcionalne zahtjeve sustava***.
- Temeljeni su na ideji scenarija.
- Služe za izlučivanje zahtjeva prema ***pogledu interakcije***.
- Pogodni su za modeliranje velikih i složenih programskih produkata.
- Jednostavni su za razumijevanje ali posjeduju i napredne značajke neophodne za ekspertne analitičare, arhitekte sustava i programere.
- Specificiraju sustave ***neovisno o načinu implementacije***.
- Mali skup konstrukcija (10% - 20%) upotrebljavamo u 80%-90% mjesta u sustavu.

DIJAGRAMI INTERAKCIJA

Dinamičke interakcije u sustavu

- Modeliranje ponašanja, *engl. behavioral modeling*
- Detaljniji razvoj i prikaz scenarija u izlučivanju, analizi i dokumentiranju zahtjeva:

- Obrasci uporabe identificiraju individualne interakcije u sustavu.
- Dodatne informacije u inženjerstvu zahtjeva uz obrasce uporabe slijede iz UML dijagrama koji pokazuju aktore uključene u interakciju, entitete (objekte, instancije) s kojima su u interakciji i operacije pridružene tim objektima.
- To su UML dinamički dijagrami interakcija:
  - sekvencijski
  - kolaboracijski.

Dijagram interakcija

- *engl. Sequence diagram*
- Specificira kontrolnu logiku scenarija opisanog obrascem uporabe
- Statična grafička reprezentacija

Interakcije

- ***Interakcija*** – skup komunikacija između objekata, uključujući sve što utječe na objekte, kao što su pozivi funkcija, stvaranje i brisanje objekata.
- Komunikacije su (djelomično) vremenski uređene!
- **Kada (zašto) modelirati interakcije?**
  - Radi definiranja kako objekti međusobno djeluju (komuniciraju)
  - Radi identifikacije sučelja (objekata/modula)
  - Radi raspodjele zahtjeva (prema dijelovima sustava)

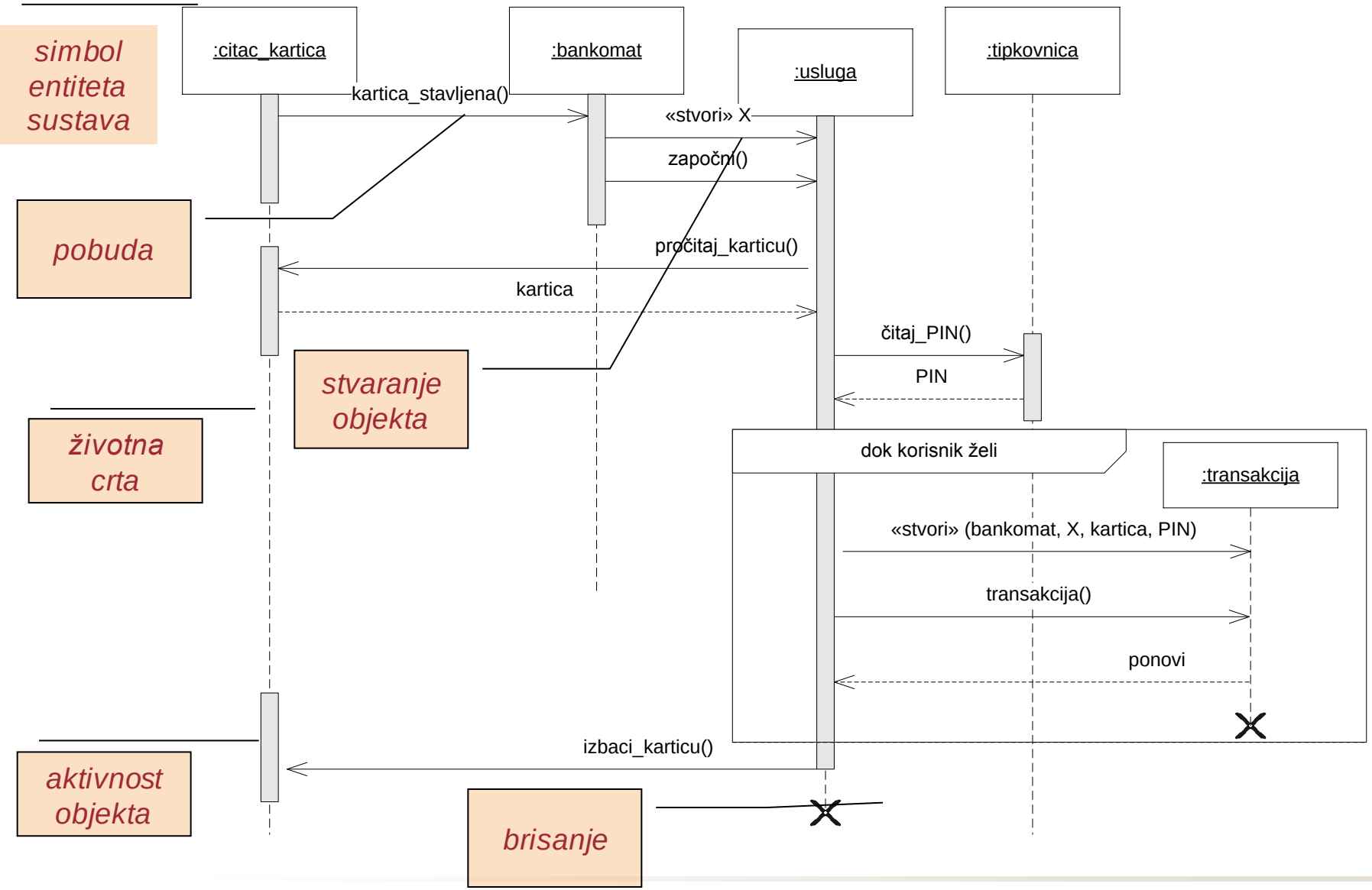
Modeliranje interakcije

- Preporuke:
  - Definirati okolinu interakcije
  - Uključi samo relevantna obilježja objekta
  - Akcije izražavaj od lijeva na desno te od vrha prema dolje
  - Aktivne objekte postavi gore-lijevo, a pasivne dolje-desno
  - Sekvencijske dijagrame (obavezno) koristi:
    - za prikaz uređenja među događajima (odvijanju akcija)
    - pri modeliranju sustava za rad u stvarnom vremenu

Sekvencijski dijagram

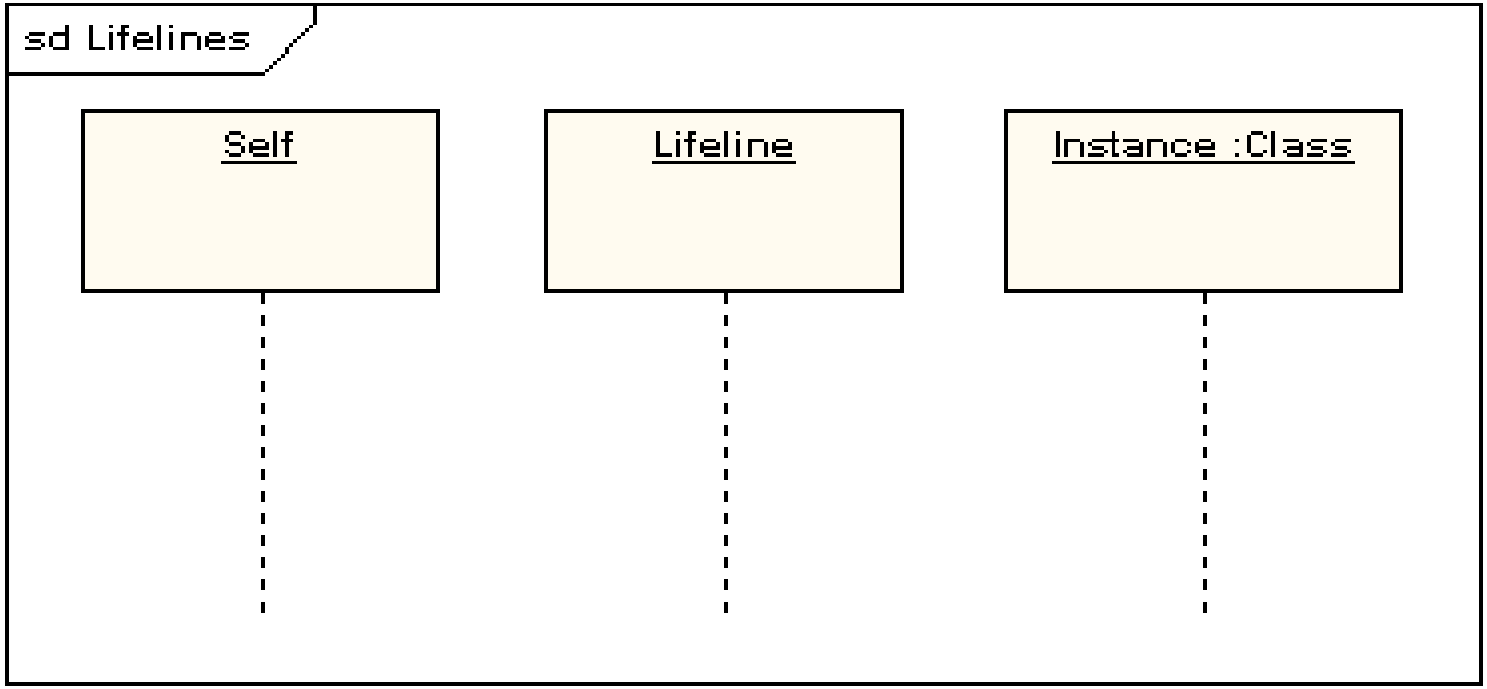
- ***Sekvencijski dijagram*** je oblik dijagrama interakcije koji pokazuje objekte kao ***životne crte*** koje idu prema dnu stranice.
- Interakcije u vremenu prikazuju se kao poruke korištenjem strelica od ***životne crte*** početnog objekta (koji pokreće interakciju/šalje poruku/poziva metodu) do životne crte odredišnog objekta (koji prima poruku…)
- Sekvencijski su dijagrami dobri za prikazivanje (identificiranje) objekata koji ***međusobno komuniciraju*** i koje poruke pokreću tu komunikaciju.
- Sekvencijski dijagrami ***nisu*** namijenjeni za prikaz ***složene proceduralne logike*** (algoritama).

Primjer:


simbol entiteta sustava
:citac_kartica
:bankomat
:usluga
:tipkovnica
kartica_stavljena()
«stvori» X
započni()
pobuda
pročitaj_karticu()
kartica
čitaj_PIN()
PIN
stvaranje objekta
životna crta
dok korisnik želi
:transakcija
«stvori» (bankomat, X, kartica, PIN)
transakcija()
ponovi
izbaci_karticu()
aktivnost objekta
brisanje

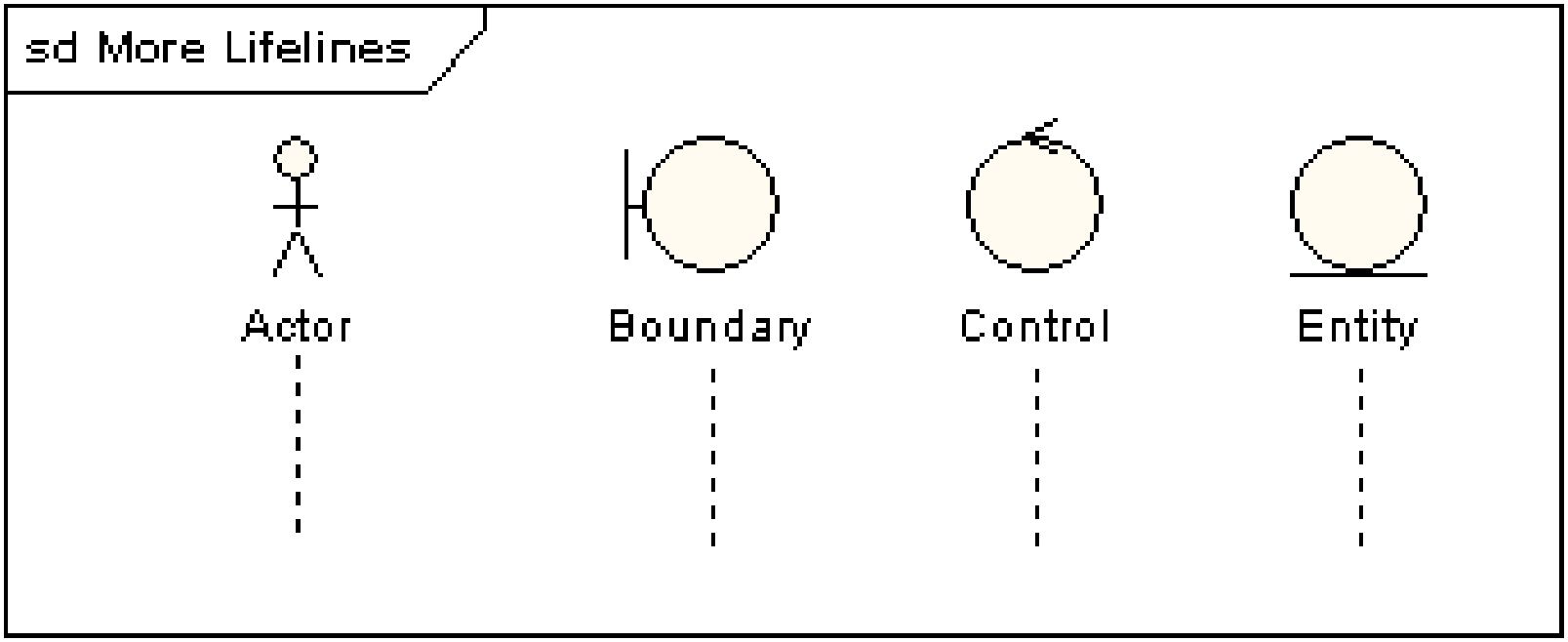
Životne linije (crte)

- *engl. Lifelines*
- Životna crta prikazuje pojedinačnog sudionika u sekvencijskom dijagramu. Obično (na početku) sadrži pravokutnik s imenom objekta (entiteta).



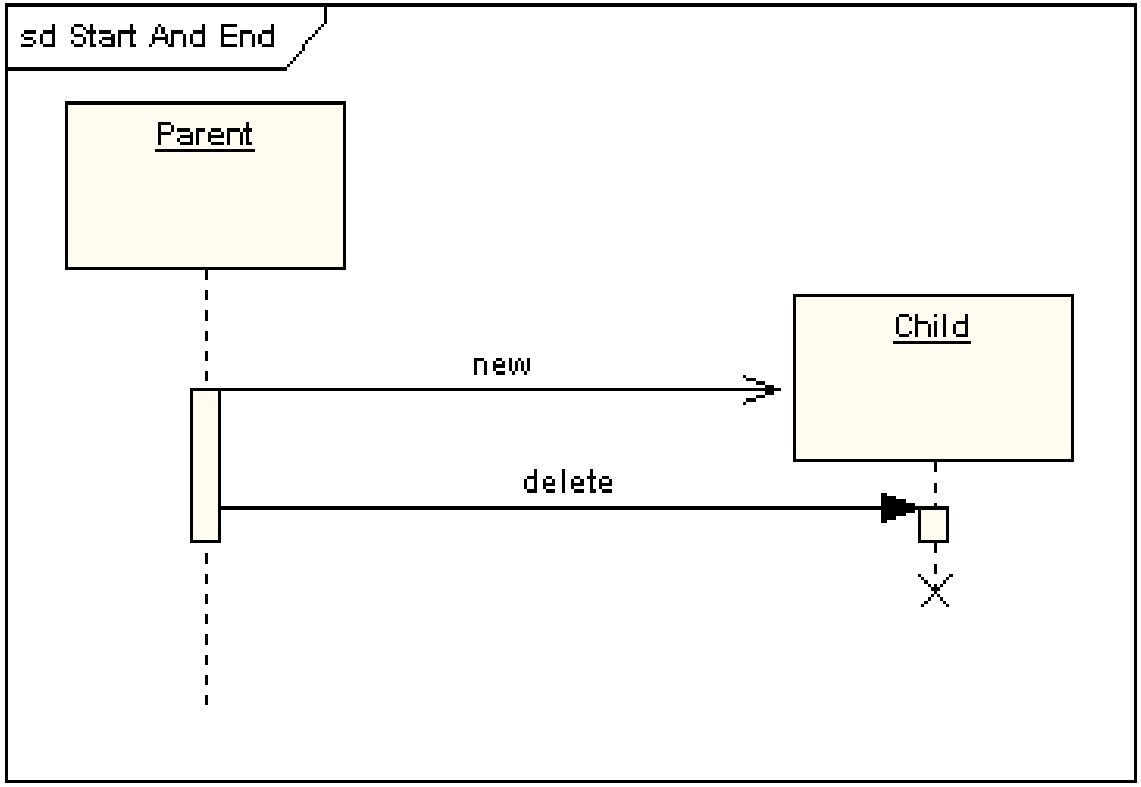
Životne linije (crte)

- Aktori se obično koriste kada sekvencijski dijagram pripada određenom obrascu uporabe.
  - Osim aktora, ostali elementi (Granični elementi i elementi upravljanja) mogu imati životnu crtu (vrlo rijetko!).



Početak i kraj životnih crta

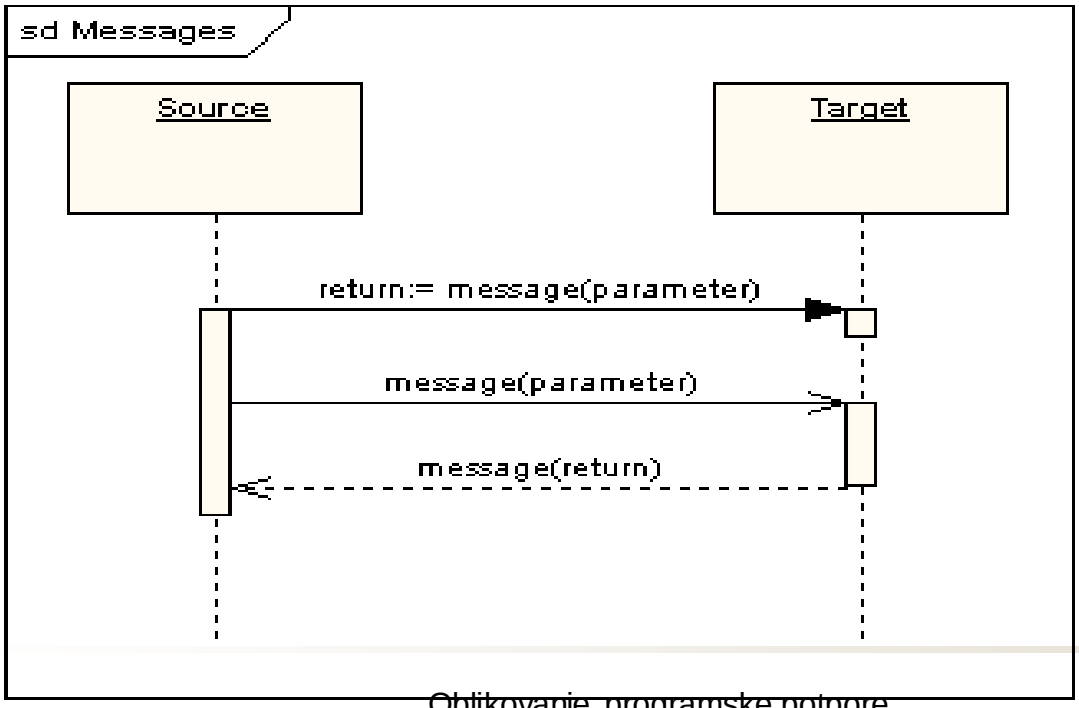
- *engl. Lifeline Start and End*
- Životna crta može biti stvorena ili prekinuta unutar jednog sekvencijskog dijagrama. Navedeni dijagram prikazuje stvaranje i brisanje objekta (nastanak i prekid životne crte).



Poruke

- *engl. Messages*

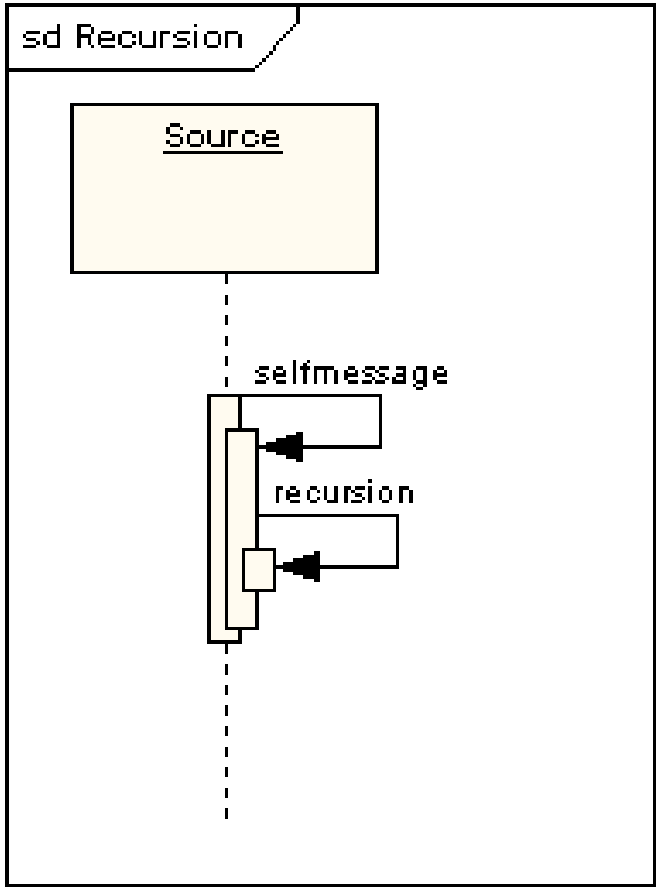
- Poruke su prikazane strelicama. Poruke mogu biti kompletne, izgubljene ili nađene; sinkrone ili asinkrone; poziv ili signal. U prikazanom dijagramu prva je poruka sinkrona (prikazana ispunjenom strelicom) i kompletna, s implicitnom povratnom porukom; druga je asinkrona (obična strelica), a treća je asinkrona povratna poruka (prikazana isprekidanom linijom). (Navedena sintaksa se često ne koristi dosljedno.)



sinkrona poruka = pošiljatelj čeka na odgovor (rezultat)

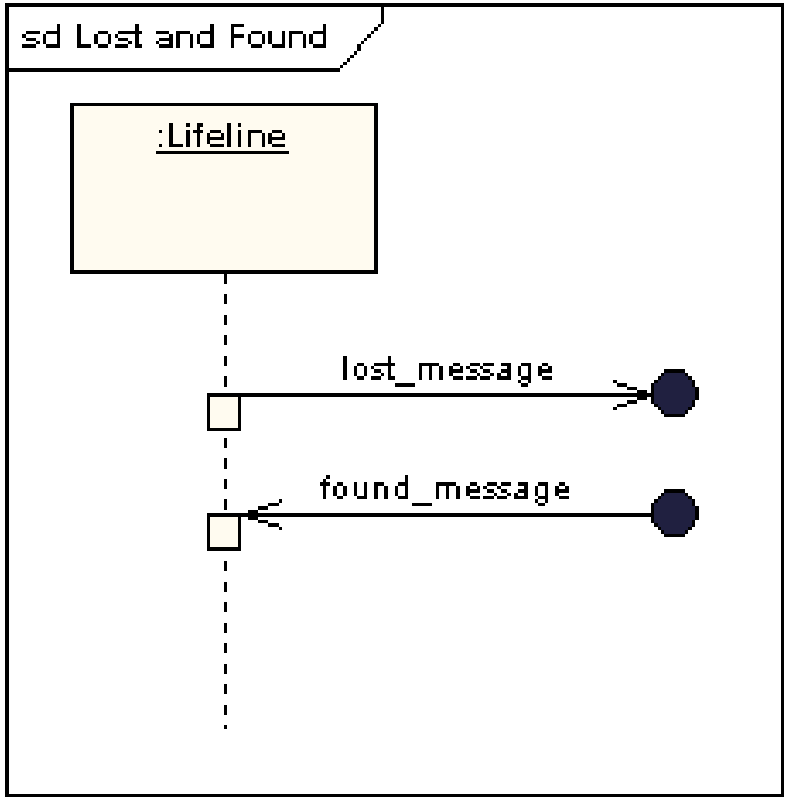
Rekurzije

- **Poruka samom sebi** - *engl. Self Message*
  - Poruka samom sebi može predstavljati rekurzivni poziv, ili poziv druge metode istog objekta.
- Poruka samom sebi:
  - različite procedure, funkcije metode u objektu;
- Rekurzija:
  - ista procedura,
  - funkcija,
  - metoda.



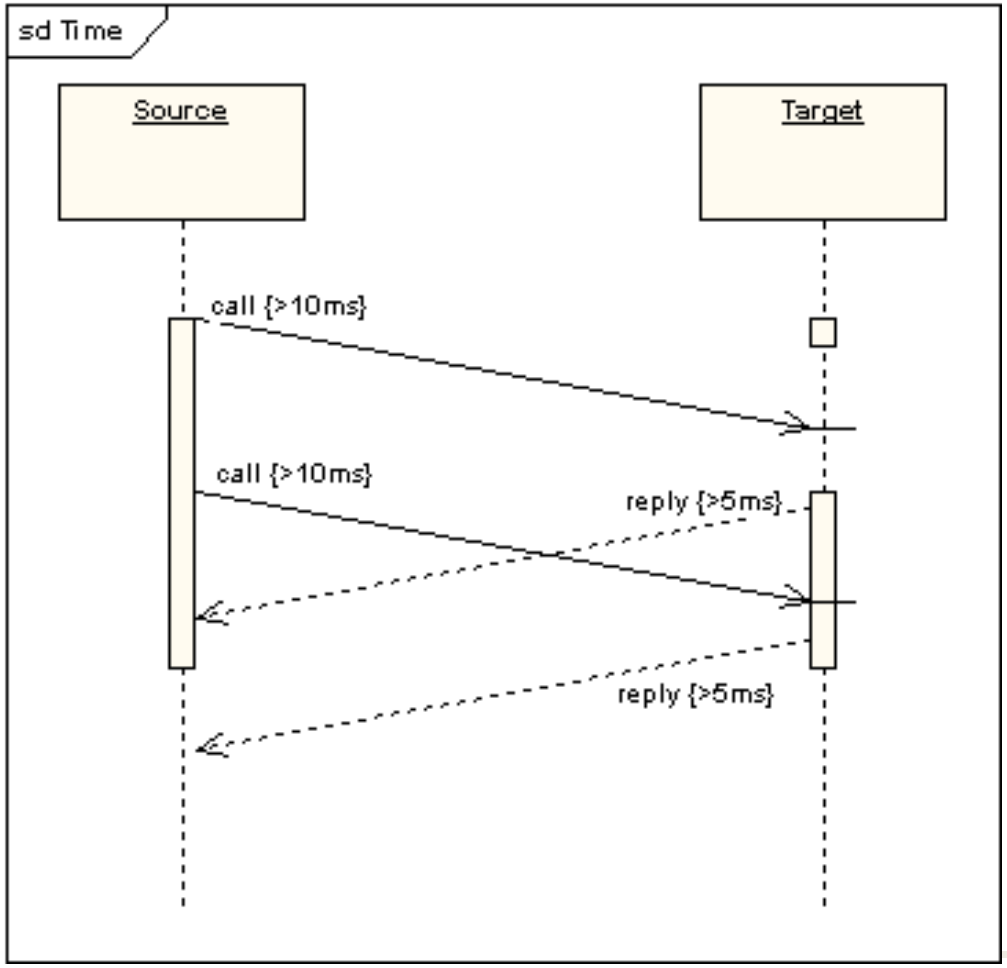
Izgubljene i nađene poruke

- ***Izgubljene poruke***
  - poslane, ali nisu stigle do odredišta;
  - poruke koje se šalju primateljima koji nisu prikazani na trenutnom dijagramu.
- ***Nađene poruke***
  - dolaze od nepoznatog pošiljatelja;
  - dolaze od pošiljatelja koji nisu prikazani na trenutnom dijagramu.



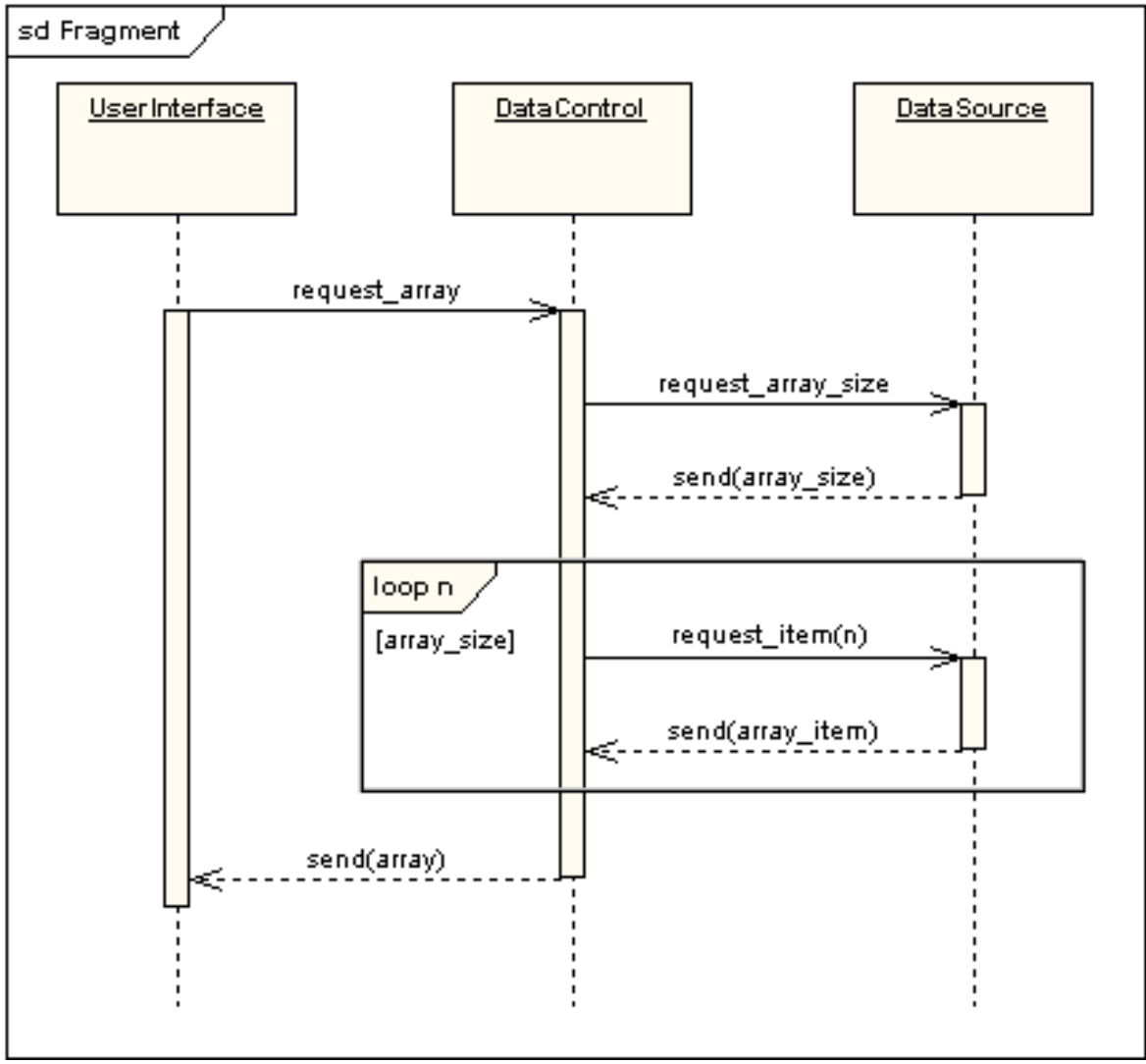
Ograničenja trajanja

- Uobičajeno se poruka prikazuje kao vodoravna crta.
- Postojanje vremenskog ograničenja obilježava se nagibom.



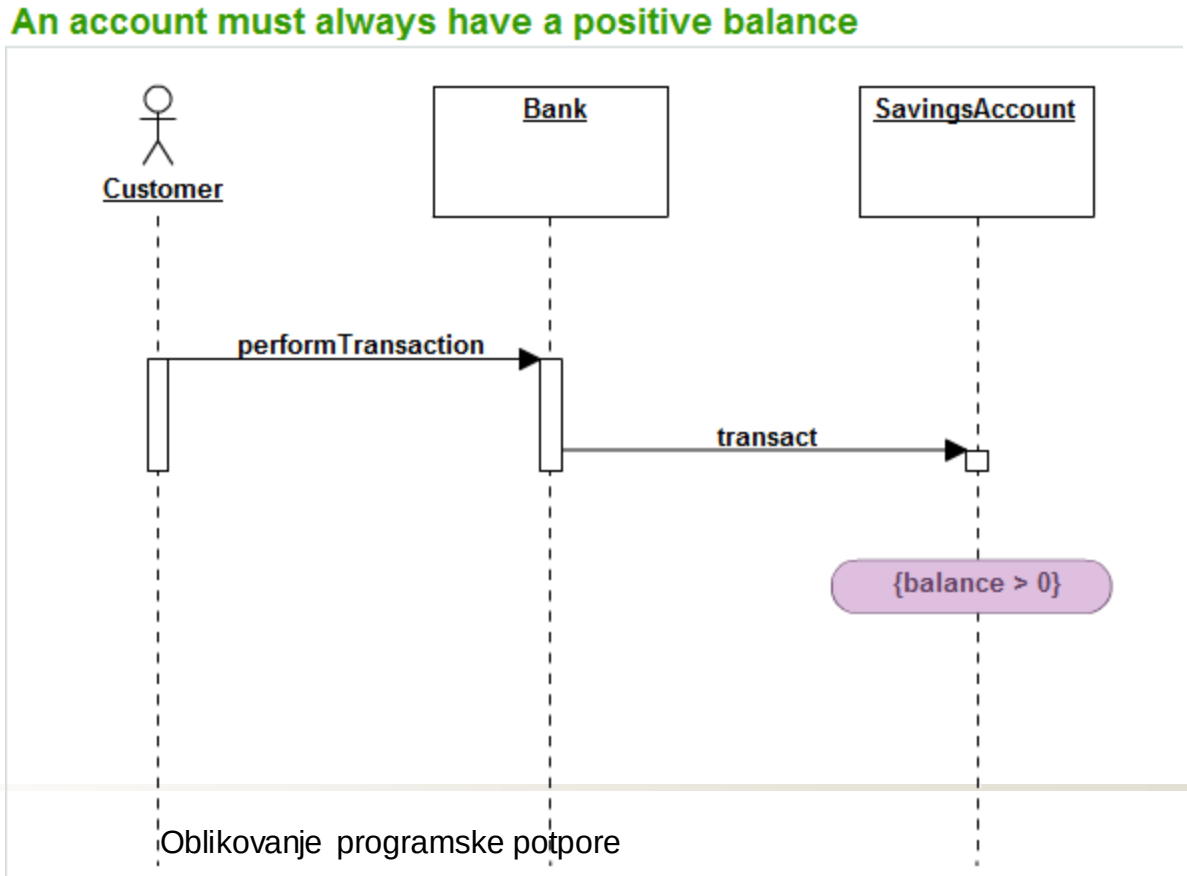
Petlja

- *engl. Loop*
- Petljom se može predstaviti niz poruka koje se ponavljaju (određeni broj puta)



Invarijante stanja

- *engl. State Invariant*
- Predstavlja ograničenja na životnoj crti na kojoj se nalazi.
  - Izračunava se pri izvođenju (*engl. runtime*) i ako nije zadovoljeno poruka se smatra nevaljanom (ona koja prethodi ograničenju).



Označavanje poveznica

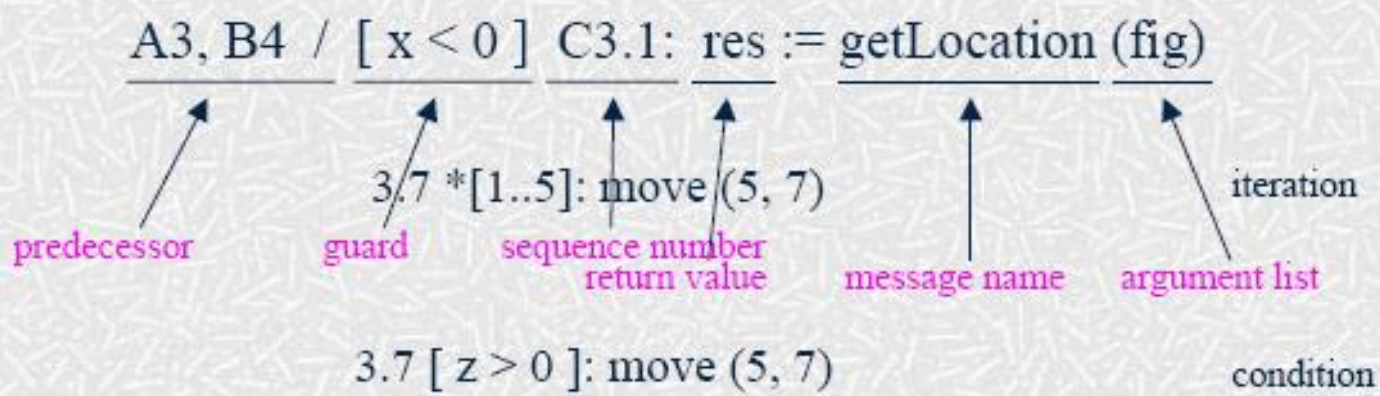
- *engl. Arrow label*

prethodnik uvjet oznaka sekvence pov. vrijed. ime poruke lista arg.

predecessor guard-condition sequence-expression return-value := message-name argument-list

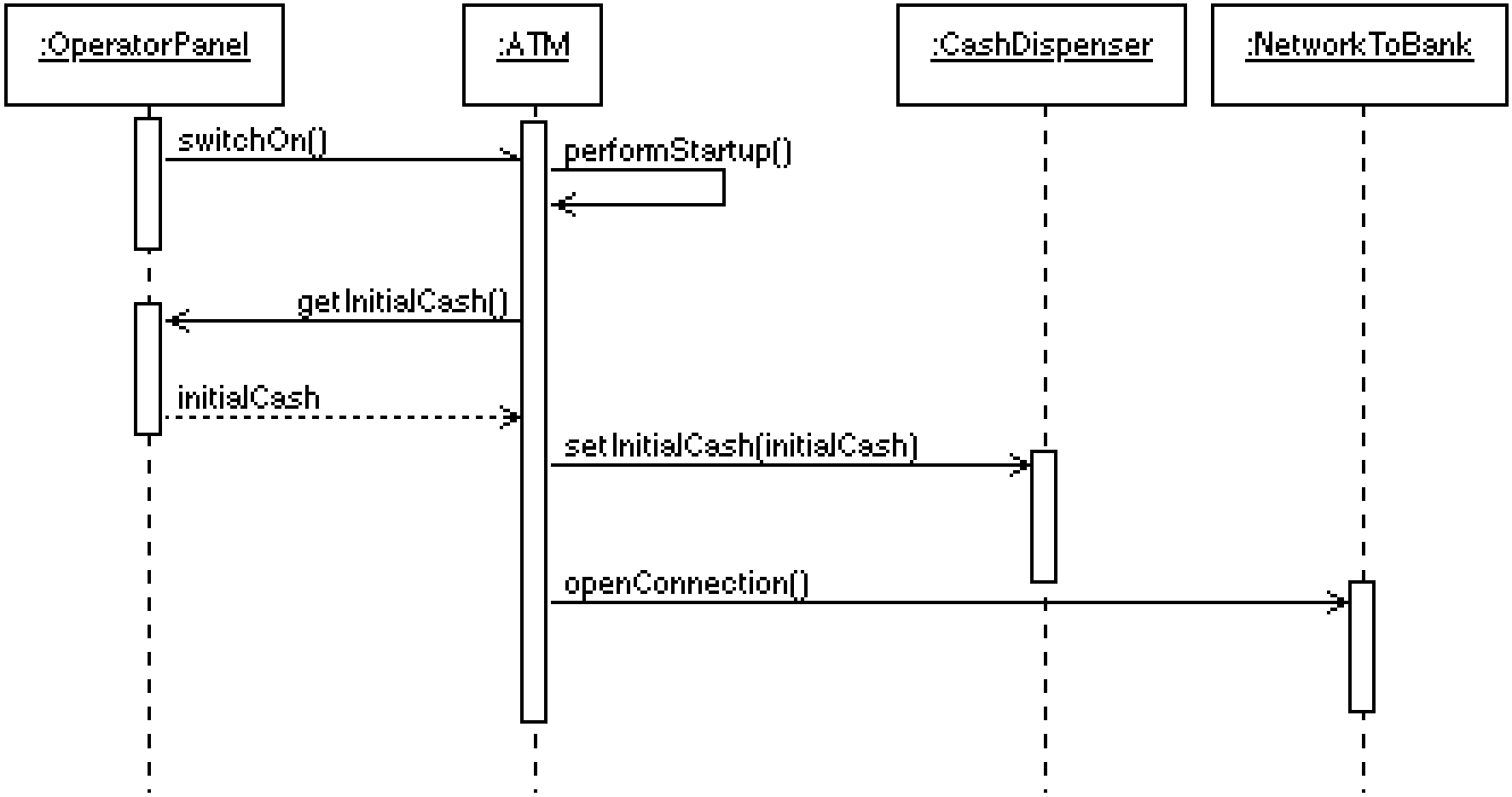
move (5, 7)

3.7.4: move (5, 7)

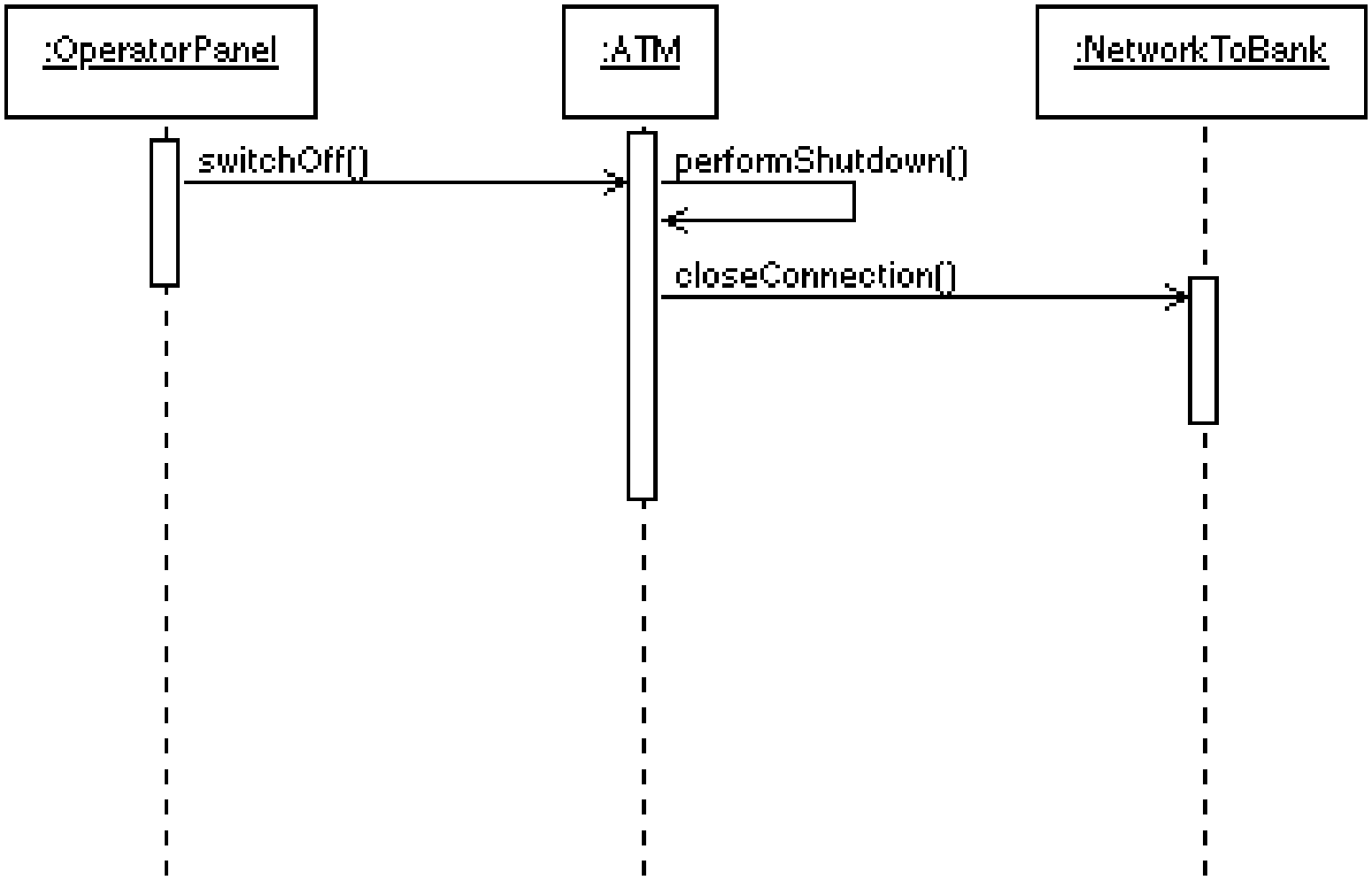


Primjer: bankomat

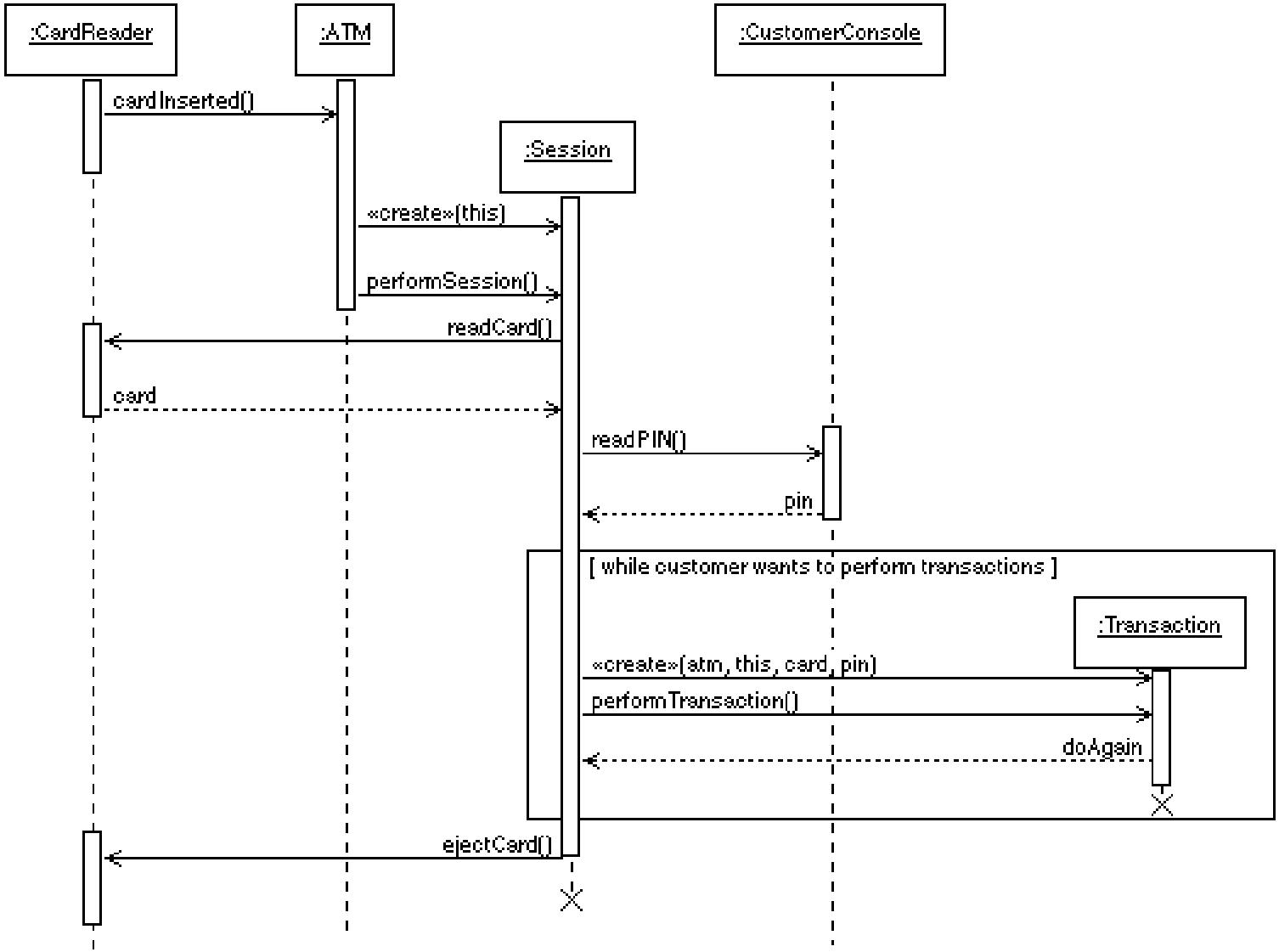
- Sekvencijski dijagram pokretanja



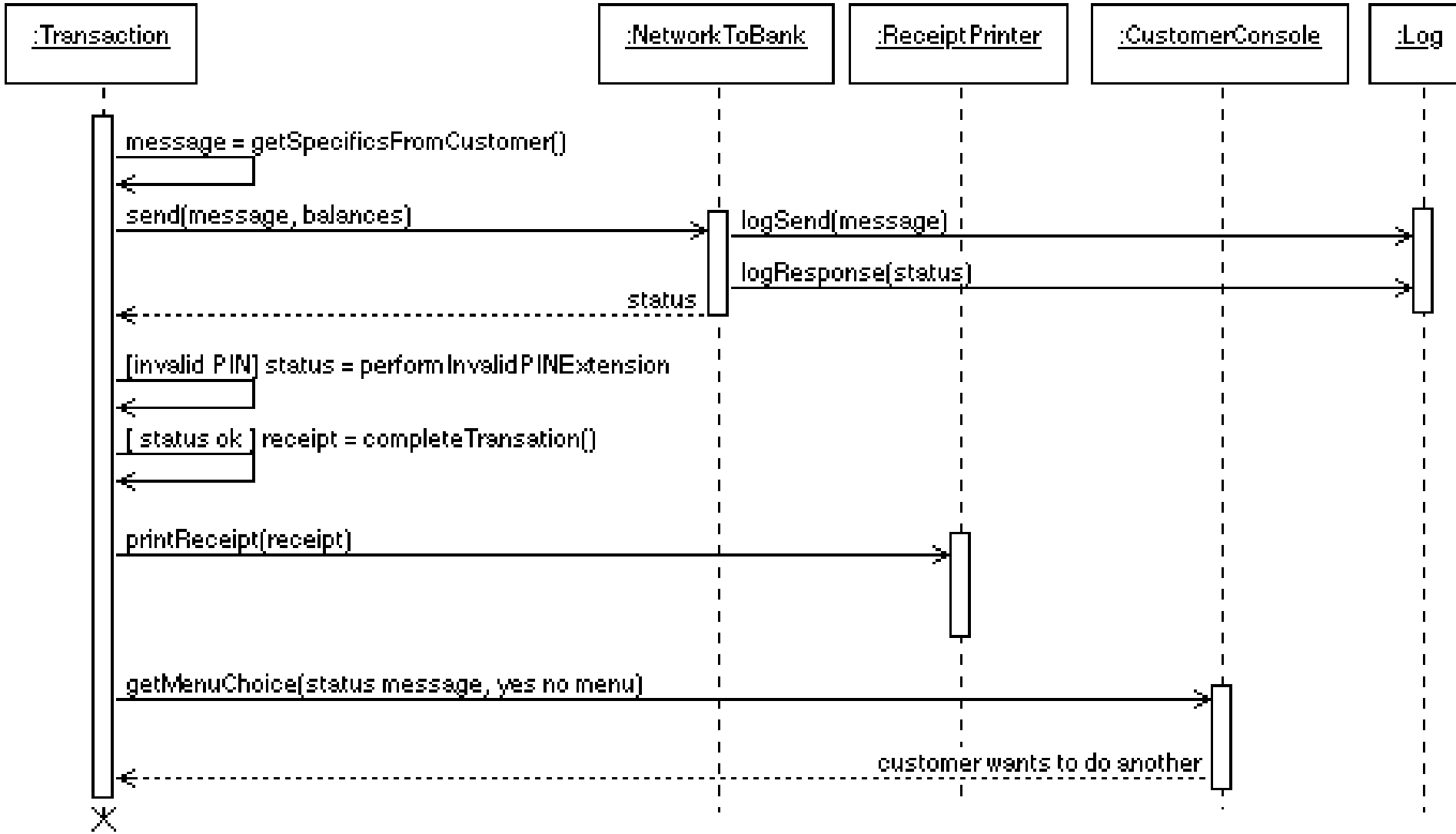
Sekvencijski dijagram gašenja



Sekvencijski dijagram usluge



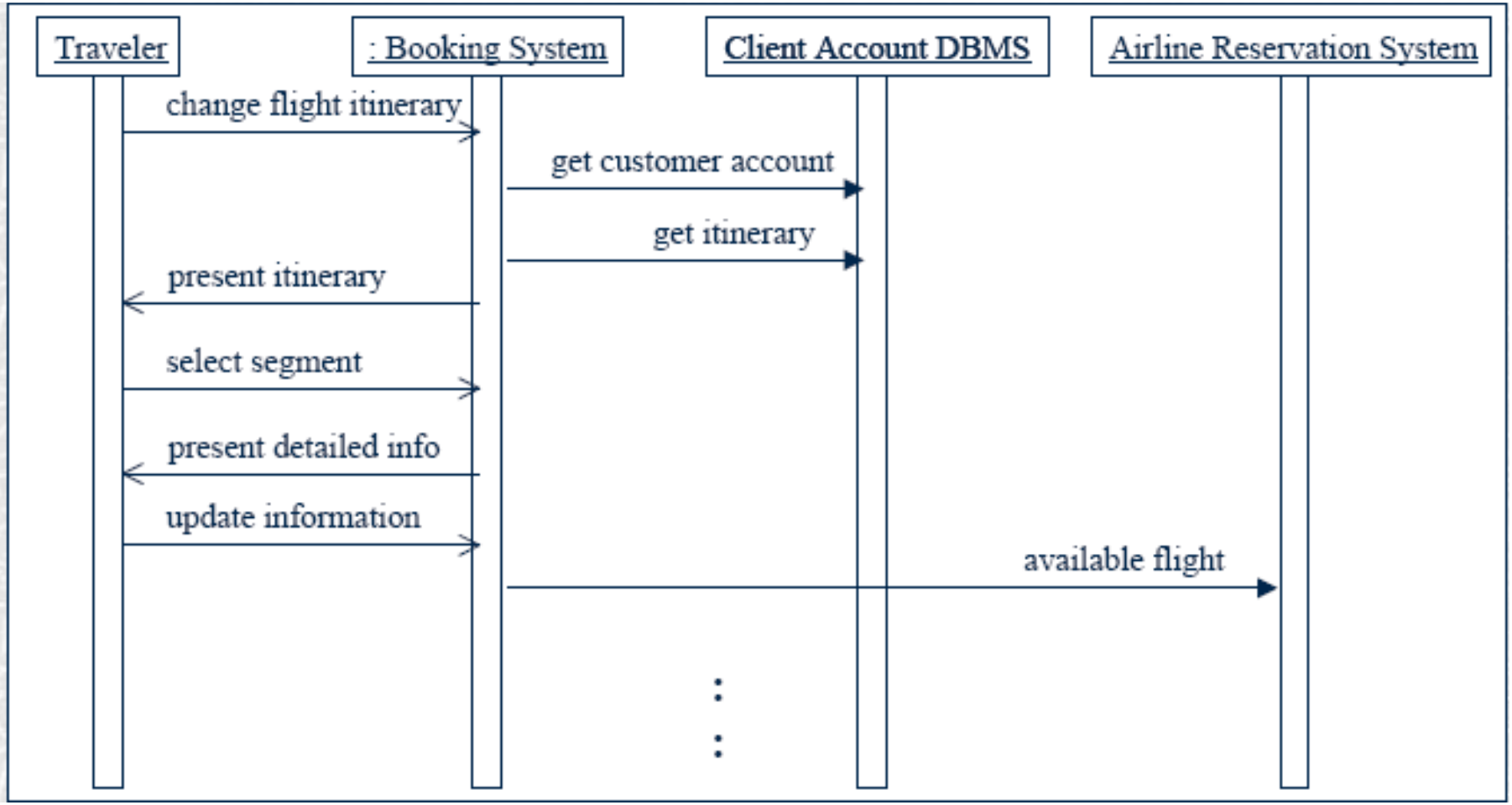
Sekvencijski dijagram transakcije



Primjer: Promjena rute leta

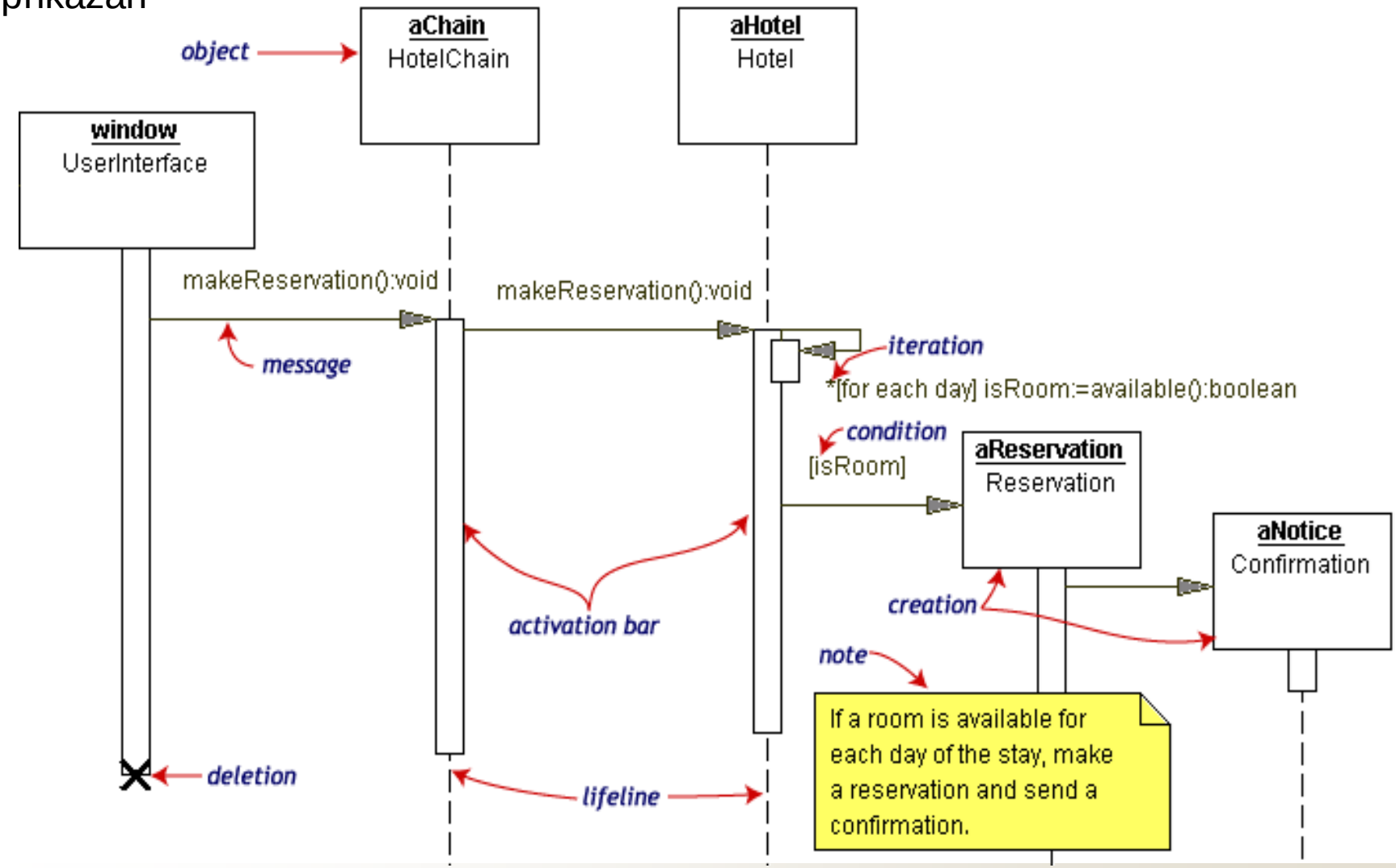
- **Aktori:**
  - putnik, baza računa klijenta (s planom puta), rezervacijski sustav avio kompanije.
- **Preduvjeti:**
  - Putnik se prijavio na sustav i odabrao opciju “promjena leta”.
- **Temeljni tijek transakcija**
  - Sustav dohvaća putnikov bankovni račun i plan puta iz baze.
  - Sustav pita putnika da odabere dio plana puta koji želi mijenjati; putnik selektira segment puta.
  - Sustav pita putnika za novi odlaznu i dolaznu destinaciju; putnik daje traženu informaciju.
  - Ako je let moguć, tada …
  - …
  - Sustav prikazuje sažetak transakcije..
- **Alternativni tijek transakcija**
  - Ako let nije moguć, tada …

Sekvencijski dijagram



Primjer: Rezervacija hotela

- Početni objekt koji započinje niz razmjena poruka je `Reservation window` nije prikazan



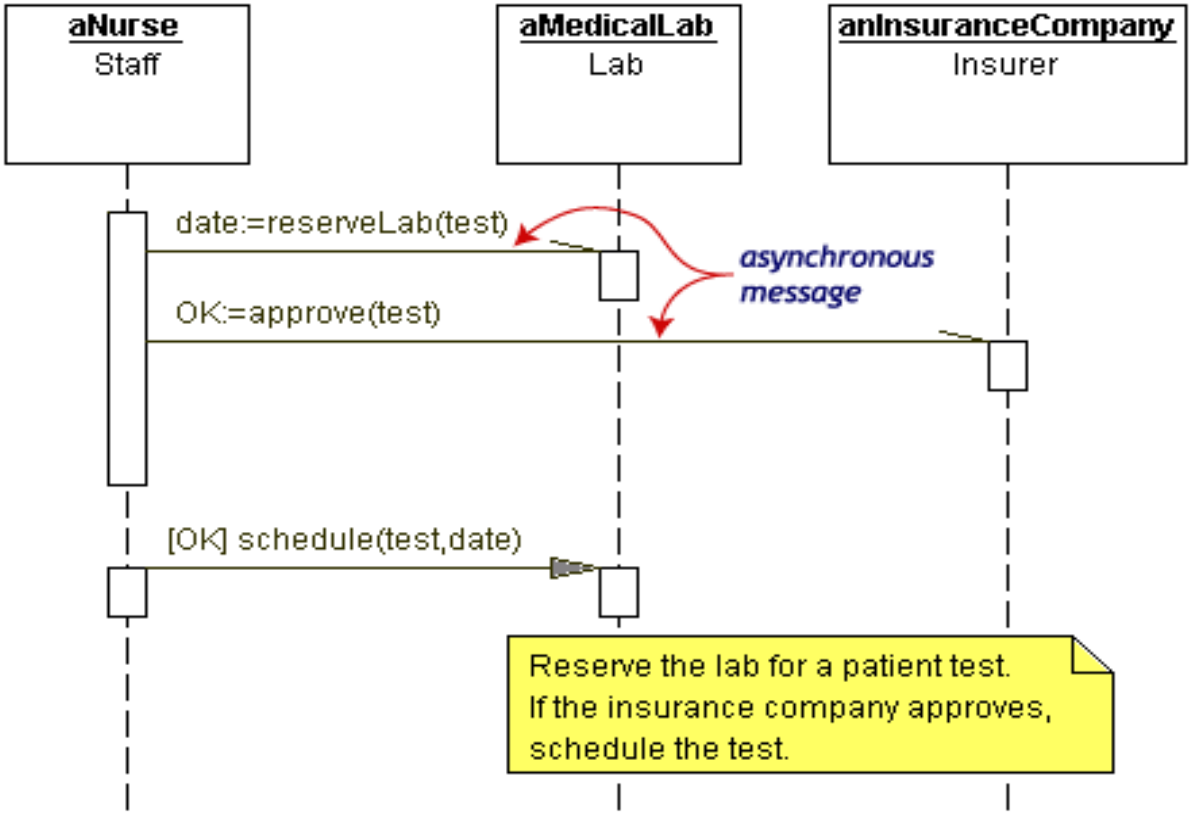
Primjer: dijagnostički test

Dvije asinkrone poruke od `aNurse`:

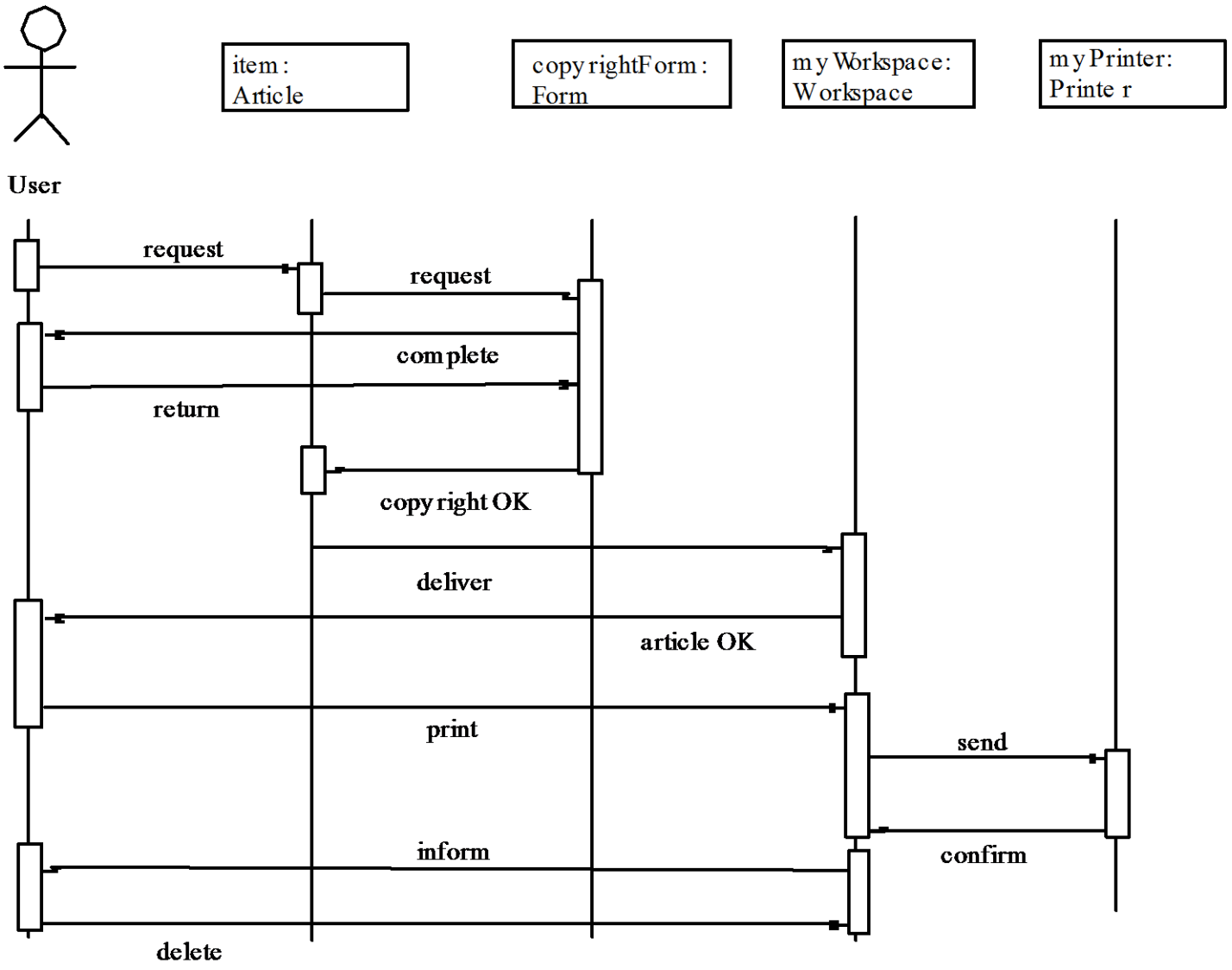
1) zahtjev prema `aMedicalLab-u` za rezervaciju termina za test
2) zahtjev prema `anInsuranceCompany` radi odobrenja

Redoslijed kojim se poruke šalji i odgovori primaju nisu bitni.

Ako `anInsuranceCompany` odobri test, tada će `aNurse` rezervirati termin za test prema datumu koje je dao `aMedicalLab`.



Primjer: LIBSYS - dijagram sekvenci za ispis članka



Sekvencijski dijagram

- ***Sekvencijski dijagram*** pogodan za opis ponašanja interakcijama
- Interakcije u vremenu prikazuju se kao poruke korištenjem strelica od ***životne crte*** početnog objekta (koji pokreće interakciju/šalje poruku/poziva metodu) do životne crte odredišnog objekta (koji prima poruku…)
- Dobri za prikazivanje i identificiranje objekata koji ***međusobno komuniciraju*** i koje poruke pokreću tu komunikaciju.
- Pomažu kod pronalaženja nedostajućih objekata
- Vremenski zahtjevan proces izgradnje
- Omogućavaju prikaz paralelnosti u sustavu
- Sekvencijski dijagrami ***nisu*** namijenjeni za prikaz ***složene proceduralne logike*** (algoritama).

Zaključci

- UML osigurava izvrsnu notaciju za opis različitih potreba u razvoju programske potpore
- Izražajan ali i složen jezik
- Zamke: mogućnost izrade nečitljivih i pogrešno interpretiranih modela
- Obrasci uporabe pogodni za opis funkcionalnih modela
- Sekvencijski dijagrami pogodni za opis dinamičkih modela

Diskusija