UML

- služi za vizualizaciju, specifikaciju, oblikovanje i dokumentiranje artefakata programske potpore
- omogućava
  - ⇒ višestruke međusobno povezane poglede
  - ⇒ poluformalnu semantiku izraženu kao metamodel
  - ⇒ jezik za opis formalnih logičkih ograničenja
- osnovni elementi:
  - ⇒ stvari
    - strukturne stvari
    - stvari ponašanja
    - stvari grupiranja
    - stvari označavanja
  - ⇒ relacije
    - ovisnosti
    - asocijacije
    - generalizacije
    - realizacije
  - ⇒ dijagrami
    - statični (*use case*, *class*, *object*, *component*, *deployment*)
      - ⇒ obrasci uporabe (*use case*) - pogled na sustav koji naglašava njegovo vanjsko ponašanje prema korisniku
        - skup koji opisuje sve moguće interakcije sustava
        - osnovni elementi:
          - ⇒ obrazac uporabe / sekvenca akcija
          - ⇒ aktor / uloga korisnika - jedan korisnik može imati više uloga; aktor može biti i vanjski sustav
          - ⇒ granica sustava
        - osnovne poveznice:
          - ⇒ pridruživanje / asocijacija - komunikacija instance aktora i instance obrasca uporabe
          - ⇒ proširenje - od proširene do osnovne uporabe (smjer strelice na dijagramu)
          - ⇒ obuhvaćanje / nužno sadrži - od osnovnog do uključenog obrasca (smjer strelice)
            - definira ponašanje uključenog u odnosu na osnovni obrazac uporabe
            - osnovni obrazac sadrži ponašanje definirano u uključenom
          - ⇒ poopćenje - odnos između općeg i specifičnog (strelica prema općem)
        - vrste:
          - ⇒ dijagram
          - ⇒ tekstovni opis
        - temelj - ideja scenarija
        - uporaba:
          - ⇒ modeliranje funkcionalnih zahtjeva sustava
          - ⇒ izlučivanje zahtjeva prema pogledu interakcije (pogodno za velike i složene programske produkte)
          - ⇒ specifikacija sustava neovisno o načinu implementacije
        - opisuje tipičnu uporabu sustava
        - mora specificirati najvažnije funkcionalne zahtjeve
        - detaljna specifikacija može dovesti do otkrivanja novih zahtjeva
        - opis značajki važnih dizajneru i naručitelju
        - jednostavan, jednoznačan i potpun
        - specifikacija pogodna za razvoj i testiranje

⇒ dijagram razreda (*class*)

- oznake vidljivosti
  - ⇒ + public - dostupno svima
  - ⇒ # protected - dostupno unutar razreda
  - ⇒ - private - dostupno podrazredima
  - ⇒ ~ packet - dostupno u paketu
  - ⇒ / derived - može se izvesti iz drugih atributa
  - ⇒ _ static
- tipovi povezivanja
  - ⇒ pridruživanje - postoji odnos između dva razreda; brojnost pokazuje broj instanci razreda
    - moguće ograničiti smjer dodavanjem strelice na jednom kraju, inače bidirekcijsko
    - opisuje odnos između instanci koji će nastupiti za vrijeme izvođenja programa
  - ⇒ agregacija - *„is part of“*
    - predstavlja odnos cijeli-dio
    - upotrebljava se kad se može tvrditi da su dijelovi dio agregata
    - propagacija - isto što i nasljeđivanje u generalizaciji, no nije implicitno
      - ⇒ implementira se operacija u agregatu tako da djeluje na dijelove
      - ⇒ značajke dijelova propagiraju natrag prema agregatu
    - crta se s rombom kod agregata
  - ⇒ kompozicija
    - jaki tip agregacije - kad se uništi agregat, uništavaju se i njegovi dijelovi
    - romb je zacrnjen
  - ⇒ generalizacija - specijalizacija superrazreda u jedan ili više podrazreda
    - generalizacijski skup - grupa generalizacija sa zajedničkim superrazredom
    - oznaka/diskriminator - kriteriji za specijalizaciju
    - opisuje odnos između razreda
    - crta se s trokutom uperenim prema zajedničkom superrazredu
  - ⇒ ovisnost - utjecaj jednog razreda na drugi
    - crta se uvijek usmjereno, bez brojnosti, linija je iscrtkana
    - tipovi: <<call>>, <<create>>, <<modifies>>
- proces razvoja
  - ⇒ istraživački model domene primjene - malo detalja; razvija se tijekom analize s ciljem razumijevanja te domene
  - ⇒ sistemski model domene
    - više detalja; modelira aspekt domene koji će biti predstavljen programskim sustavom
    - može izostaviti razrede potrebne u izgradnji cjelovitog programskog sustava
    - izgrađuje se i koristi ovisno o skupu razreda koji modeliraju korisničko sučelje i skupu razreda koji modeliraju arhitekturu sustava
  - ⇒ model sustava
    - najviše detalja; uključuje razrede objekata potrebnih u izgradnji arhitekture sustava i korisničkog sučelja
    - sistemski model domene
    - razredi koji modeliraju korisničko sučelje
    - razredi koji modeliraju arhitekturu sustava
    - pomoćni razredi

- odgovornost - nešto što sustav mora izvršiti
  ⇒ veća razina apstrakcije od operacije
  ⇒ kategorije
    - postavljanje i dohvaćanje vrijednosti atributa
    - kreiranje i inicijalizacija novih instanci
    - upis i čitanje iz trajne pohrane podataka
    - uništavanje instanci
    - dodavanje i brisanje veza pridruživanja između instanci
    - kopiranje, konverzija, preslikavanje, prijenos i izlaz podataka
    - izračunavanje numeričkih vrijednosti
    - navigacija i pretraživanje
    - ...
- CRC kartice (*class-responsibility-collaboration*) - za oblikovanje dijagrama razreda na papiru
  ⇒ za svaki razred se na posebnu karticu popišu njegove odgovornosti i atributi
  ⇒ ako nije moguće sve odgovornosti ispisati na jednoj kartici, vjerojatno treba odijeliti razred u dva međusobno povezana razreda

⇒ dijagram objekata (*object*)
- prikazuje instance i veze u jednom trenutku izvođenja sustava
- generira se iz dijagrama razreda

⇒ komponentni dijagram (*component*)
- komponenta - zamjenjiv, ponovno iskoristiv dio programskog kôda
  ⇒ vrste: izvorni kôd, binarni kôd, statičke ili dinamičke knjižnice, izvršne (*exe*), tablice, datoteke, baze podataka...
  ⇒ UML komponente
    - fizičke modularne zamjenjive jedinke s dobro definiranim sučeljem
    - može im se pristupati samo kroz sučelja
    - mogu sadržavati druge komponente (prikaz interne strukture)
    - mogu imati definirane portove (točka interakcije s okolinom)
    - prikaz instance: *ime_komponente: tip_komponente*
- komponentno zasnovano programsko inženjerstvo integrira sustav:
  ⇒ višestrukom uporabom postojećih komponenti
  ⇒ uporabom gotovih komponenti (*commercial-off-the-shelf*)
  ⇒ modificiranjem komercijalnih komponenti (*modified-off-the-shelf*)
- organizacija i međuovisnost između implementacijskih komponenata programske potpore
- dio specifikacije arhitekture programske potpore
- oblikuju ih arhitekti programske potpore i programeri
- namjena: prikaz komponenata, interne strukture i odnosa prema okolini
- sučelje
  ⇒ imenovan skup javno vidljivih atributa i apstraktnih operacija (implementaciju osigurava komponenta)
  ⇒ tipovi:
    - ponuđeno - usluga koja se nudi, koju ostvaruje razred ili komponenta
    - zahtijevano - ono što je potrebno komponenti za rad
  ⇒ tipovi konektora
    - spojnica - povezuje dvije komponente (*ball and socket*, *lollipop*)
    - delegacija - povezuje sučelje komponente s internom strukturom
- ovisnost - relacija koja se upotrebljava kad jedna komponenta zahtijeva uporabu druge radi potpunog ostvarenja implementacije (crtkana strelica od ovisne komponente prema onoj o kojoj ovisi)

- stereotipi
  - ⇒ *executable* - može se izvršavati
  - ⇒ *library* - statička ili dinamička biblioteka
  - ⇒ *file* - datoteka
  - ⇒ *document*
  - ⇒ *script*
  - ⇒ *source* - datoteka s izvornim kodom

⇒ dijagram razmještaja (*deployment*)
- opis topologije sustava, fokus na odnosu sklopovskih i programskih dijelova
- prikaz sklopovskih komponenti, komunikacijskih putova te smještaja i izvođenja programskih artifakata
- sklopovske komponente:
  - ⇒ čvorovi:
    - uređaj (<<device>>) - stvarni ili virtualni; procesna jedinka
    - okolina izvođenja (<<execution environment>>) - programski sustav (npr. OS, interpreter, virtualni stroj)
  - ⇒ spojevi - komunikacijski putovi
- programske komponente (artifakti) - implementirani moduli i podaci; prikaz odnosa između komponenti pomoću ovisnosti

⇒ dijagram paketa (*package*)
- paket
  - ⇒ mehanizam organiziranja elemenata u grupe
  - ⇒ povezuje semantički bliske elemente koji imaju tendenciju zajedničkih promjena
  - ⇒ vidljivost:
    - public (+) - sadržaj vidljiv svim paketima koji ga koriste
    - protected (#) - mogu ga vidjeti samo djeca tog paketa
    - private (-) - nije vidljiv izvan paketa
- prikaz dekompozicije modela u organizirane grupe i njihove međuodnose

⇒ dijagram profila (*profile*)
- proširenje jezika za stvaranje novih dijagrama
- statički dijagram strukture, prikazuje proširenje postojećih ili nove elemente modeliranja
- definiranje korisničkih stereotipa, vrijednosti, ikona za specifična područja primjene, tehnologije ili metode
- UML profil
  - ⇒ skup predefiniranih stereotipa (ne proširuje UML)
  - ⇒ pojednostavljena primjena UML-a u drugim domenama primjene

⇒ dijagram složene strukture razreda (*composite*)
- opis interne strukture razreda
- opis kolaboracija koje struktura razreda omogućuje
- interni dijelovi i sučelja za međusobnu interakciju dijelova i interakciju s vanjskim svijetom te konektori

- dinamični (*sequence*, *collaboration*, *statechart*, *activity*) - modeliranje ponašanja

⇒ uporaba:
- definiranje međusobnog djelovanja objekata
- identifikacija sučelja objekata ili modula
- raspodjela zahtjeva prema dijelovima sustava

⇒ preporuke:
- definirati okolinu interakcije
- uključiti samo relevantna obilježja objekta
- izražavanje akcija od gore prema dolje i slijeva nadesno
- aktivni objekti smješteni gore lijevo, a pasivni dolje desno

⇒ sekvencijski (*sequence*)
- uporaba:
  ⇒ prikaz odvijanja akcija
  ⇒ modeliranje sustava za rad u stvarnom vremenu
  ⇒ prikazivanje objekata koji međusobno komuniciraju
- nisu namijenjeni za prikaz algoritama
- elementi:
  ⇒ životne linije - prikaz objekata
  ⇒ poruke (uključuje rekurzije, izgubljene i nađene poruke) - prikaz interakcija u vremenu
    - kompletne, izgubljene ili nađene
    - sinkrone (puna strelica; pošiljatelj obično čeka odgovor) ili asinkrone (obična strelica - →; asinkrona povratna poruka se prikazuje isprekidanom linijom)
    - poziv ili signal
  ⇒ petlja (niz poruka koje se ponavljaju)
  ⇒ invarijante (uvjeti) stanja - ograničenje na životnoj crti na kojoj se nalazi; izračunava se pri izvođenju i, ako nije zadovoljeno, poruka je nevaljana

⇒ dijagram komunikacije (*communication*, *collaboration*)
- definira uloge instanci tijekom obavljanja nekog zadatka
- ne prikazuje vremenske odnose
- modelira upravljački tok - specificira tijek komunikacije između instanci tijekom suradnje
- preporuke:
  ⇒ ime treba prikazivati namjenu
  ⇒ minimizirati presijecanje linija
  ⇒ samo nužni elementi
  ⇒ važni elementi u centru dijagrama
  ⇒ sistematičnost
  ⇒ jedan dijagram prikazuje samo jedan upravljački tok

⇒ dijagram stanja (*statechart*, *state machine diagram*)
- dinamičko ponašanje jednog objekta u vremenu
  ⇒ pogodno za opis značajnijeg dinamičkog ponašanja i diskretnog ponašanja, te za modeliranje događajima poticanog ponašanaj sustava
  ⇒ promatranje objekta izolirano od ostatka sustava
  ⇒ izlaz ovisi o ulazima i povijesti
- sekvenca stanja objekta i prijelazi utemeljeni na događajima
- tri grupe aktivnosti u nekom stanju:
  ⇒ entry - pri ulasku u stanje
  ⇒ do - dok je objekt u tom stanju
  ⇒ exit - na izlasku iz stanja
- jedno početno i više mogućih krajnjih stanja
- prijelazi - inicirani događajima i uvjetima: [uvjet] akcija; mogu prenositi parametre
  ⇒ vrste
    - interakcija: asinkroni prijem signala (*signal*) ili sinkroni poziv objekta (*call*)
    - vremenski (*time*): istek vremenskog intervala, konkretan trenutak, takt...
    - ispunjeni uvjeti (*change*)
  ⇒ trajanje izvođenja je 0 i ne može se prekinuti
  ⇒ događaji - pokreću prijelaz
  ⇒ uvjet - izraz koji kada je ispunjen dopušta odvijanje prijelaza (uz pojavu događaja)
  ⇒ akcija - operacija koja se odvija
- Mealyjev automat - izlaz je funkcija ulaza i stanja
- Mooreov automat - izlaz je funkcija stanja

- značajke:
  ⇒ hijerarhijsko modeliranje stanja - prikaz složenih problema
  ⇒ modeliranje paralelnih (ortogonalnih) stanja - višestruka perspektiva istog objekta, istodobno reagiraju na iste događaje
  ⇒ aktivnosti u stanjima
  ⇒ akcija pri ulasku i izlasku iz stanja
    - prioritet akcija
      ⇒ ulaz - izvana prema unutra
      ⇒ izlaz - iznutra prema van
  ⇒ uvjetna grananja
    - statičko uvjetovanje - uvjeti poznati prije grananja
    - dinamičko uvjetovanje - uvjeti se izračunavaju pri izlasku iz stanja
- povijest - mogućnost povratka na prethodno stanje
  ⇒ povijest (*shallow history*) - povratak na posljednje stanje na istoj razini
  ⇒ duboka povijest (*deep history*) - povratak na posljednje stanje, neovisno o razini
- jedno stanje može imati više konkurentnih podstanja (višestruka perspektiva jednog objekta, smanjuje broj stanja)
- interakcija između područja - pomoću dijeljenih varijabli
- račvanje i skupljanje paralelnih prijelaza - opis prijelaza u i iz ortogonalnih paralelnih regija

⇒ dijagram aktivnosti (*activity*)
- semantika zasnovana na Petrijevim mrežama (UML 2)
  ⇒ povećana prilagodljivost modeliranju različitih tipova tijeka
  ⇒ razdvajanje od dijagrama stanja
  ⇒ opis tijeka izvođenja i ponašanja sustava
- pogodni za:
  ⇒ prikaz proceduralnog tijeka procesa - prikaz ponašanja koje ne zavisi o velikom broju vanjskih događaja ili kada postoje definirani koraci bez prekida
  ⇒ prikaz tijeka objekata između koraka
  ⇒ modeliranje poslovnih zahtjeva procesa
  ⇒ prikaz paralelnosti
  ⇒ pojašnjenje aktivnosti
- primjena:
  ⇒ opis modela toka upravljanja ili toka podataka
  ⇒ opis poslovnog modela u kojem želimo modelirati tijek zadataka i poslova
  ⇒ oblikovanje tijeka obrazaca uporabe i tijeka između različitih obrazaca uporabe
  ⇒ oblikovanje detalja operacija i algoritama
- ne primjenjuju se za modeliranje događajima poticanog ponašanja
- modeliranje toka upravljanja - novi korak se poduzima nakon završenog prethodnog, neovisno o stanju ulaza u tom trenutku (*pull* način djelovanja)
- modeliranje toka podataka - sljedeći korak se poduzima kada su svi ulazni podaci dostupni (*push* način djelovanja)
- aktivnost - više čvorova i veza koji predstavljaju odgovarajući slijed zadataka
  ⇒ modelira ponašanje kao niz akcija
  ⇒ modeliranje izvođenja uporabom znački
  ⇒ particije - podjela dijagrama aktivnosti, ne utječe na tijek odvijanja aktivnosti
  ⇒ podaktivnost
    - funkcionalna dekompozicija, ugnježđenje drugog dijagrama aktivnosti
    - ulaskom u nju se pokreće početna akcija, po završetku izvođenja nastavlja se izvoditi prethodna aktivnost
- akcija - kratkotrajno, neprekidivo ponašanje unutar čvora akcije
- uvjeti - mogu biti definirani za sve aktivnosti i akcije; prije i poslije izvođenja

- značke (*tokens*)
  ⇒ dio semantike bez grafičkog prikaza
  ⇒ može predstavljati upravljački tijek, objekt ili podatak
  ⇒ kretanje vezama od izvorišta prema odredištu ovisno o:
    - ispunjenim uvjetima izvornog čvora
    - postavljenim uvjetima veza (*edge guard*)
    - preduvjetima ciljnog čvora
- elementi:
  ⇒ čvorovi:
    - čvorovi akcije - nedjeljive aktivnosti
      ⇒ započinje izvođenje kad:
        - postoji odgovarajući broj znački na svim ulazima
        - su zadovoljeni svi lokalni preduvjeti
      ⇒ nakon izvođenja se provjerava zadovoljenje uvjeta i značke se prosljeđuju na izlaze
      ⇒ tipovi:
        - obrade osnovnih operacija, pozivanje složenih aktivnosti ili ponašanja (*call action*)
        - komunikacijske akcije:
          - slanje signala - asinkroni signal
          - prihvaćanje događaja - čekanje na neki događaj
          - vremenski događaj - definiran vremenskim izrazom
      ⇒ manipuliranje objektima
    - upravljački čvorovi - upravljanje tijekom (početak i završetak, odluke, paralelnost)
      ⇒ uvjetovanje toka aktivnosti može biti u točkama odlučivanja ili na račvanju (sažetije)
    - objektni čvor - prikaz objekata u aktivnostima
      ⇒ raspoloživost u danoj točki aktivnosti
      ⇒ označen imenom razreda, predstavlja instancu
      ⇒ ulazne i izlazne veze opisuju kretanje objekta unutar aktivnosti
      ⇒ stvaraju i upotrebljavaju akcijski čvorovi
      ⇒ kada primi značku, nudi ju na svim izlaznim vezama koje se natječu za nju (značku dobiva prva veza koja ju je spremna prihvatiti)
      ⇒ međuspremnici - pohrana objektnih znački
      ⇒ priključnice (*pins*)
        - jedan ulaz ili izlaz prema čvoru akcije
        - jednostavniji grafički prikaz dijagrama aktivnosti s većim brojem objektnih čvorova
  ⇒ veze
    - upravljački tijek - tijek upravljanja aktivnostima
    - tijek objekta - tijek objekta kroz aktivnosti
- dinamički paralelizam - aktivnost se može izvoditi više puta paralelno
  ⇒ unutar simbola aktivnosti može se označiti brojnost (maksimalni broj paralelnog izvođenja)
  ⇒ nakon završetka svih paralelnih aktivnosti inicira se prijelaz na novi korak
- objekt kao uvjet prijelaza - kada je potrebno prijelaz s jedne na drugu aktivnost uvjetovati postojanjem određene vrste objekta
  ⇒ u prijelaz se umeće simbol, stanje u kojem nema nikakve aktivnosti
  ⇒ tok se nastavlja nakon zadovoljenja uvjeta
  ⇒ modelira se iscrtkanim strelicama

⇒ dijagram pregleda interakcije (*interaction overview*)
- kombinacija dijagrama aktivnosti i sekvencijskog dijagrama - dijagram aktivnosti s dijelovima sekvencijskih dijagrama i kontrolom tijeka

⇒ vremenski dijagram (*timing*)
- za izričit prikaz stvarnih vremena - točniji zapis sekvencijskog dijagrama; prikazuje promjene stanja na liniji života u vremenu
- pogodan za primjenu u sustavima za rad u stvarnom vremenu

- uporaba (svi pogledi koriste obrasce uporabe):

⇒ logički pogled (*design view*):
- obrasci uporabe
- dijagram razreda
- dijagram aktivnosti
- komunikacijski dijagram
- sekvencijski dijagram

⇒ razvojni pogled (*development view*) i procesni pogled (*process view*):
- komunikacijski dijagram
- sekvencijski dijagram
- dijagram razreda
- dijagram stanja
- komponentni dijagram
- dijagram razmještaja

⇒ implementacijski pogled (*deployment view*):
- dijagram paketa
- dijagram razmještaja