

UML–keel

Kasutusmallid

Modelleerimine, OOA(nalysis) ja OOD(esign).
UML teke, loojad, versioonid, üldised võimalused
Kasutusmall, tegija, stsenaarium.

Mudel

Mudel on reaalsuse lihtsustatud, üldistatud esitus.

- Mudel peab aitama
 - nähtust paremini mõista;
 - tegevusi planeerida;
 - ennast (st mudelit) realiseerida.
- Mudel on kvaliteetne, kui ta:
 - kirjeldab täpselt uuritavat süsteemi;
 - on arusaadav teisele inimesele (mitte ainult modelleerijale);
 - on töödeldav arvutiga (meie kontekstis).

Modelleerimiskeeel

- **Modelleerimiskeeel** (*modeling language*) on (graafiline) vahend, mida kasutatakse erinevate mudelite - kavandite loomiseks ja esitamiseks.
- Tavalist **keelt** saab samuti kasutada mudelite ja kavandite kirjeldamiseks.
- **Modelleerimine** on tarkvara kavandamine enne kodeerimist – vajalik suuremahulise tarkvara puhul, abiks keskmise- ja väikesemahulise tarkvara juures.

Milleks modelleerida?

- Luua struktuur probleemi lahendamiseks.
- Katsetada erinevate võimalike lahenduste uurimiseks ja sobiva leidmiseks.
- Abstrahheerida keerukuse vähendamiseks – detailid välja!.
- Vähendada äri lahenduste turule jõudmise aega.
- Vähendada arenduskulusid.
- Hallata ja vähendada vigadest tekkivaid riske.
- ... jne

OO-modelleerimismeetodid

- Tekkisid 1980-te aastate keskel peale OO-keelte tulekut (varem struktuursed meetodid)
- 1988 – 1994 palju OOA ja OOD raamatuid
- Erinevad meetodid ja märgisüsteemid:
 - Booch'i meetod;
 - Rumbaugh meetod (*object modeling technique – OMT*);
 - Jacobson'i meetod (*object-oriented software engineering – OOSE*);
 - Coad'i and Yourdon'i meetod;
 - Wirfs-Brock'i meetod; ...

OO arendusmudel.Näide (1)

G. Boochi poolt on 1991 välja pakutud arendusmudel järgmistest sammudest:

- analüüsi piisavalt, et eraldada peamised klassid ja seosed, mis selles probleemis on;
- kavanda veidi, et teada saada, kas klassid ja seosed on ka praktiliselt realiseeritavad;
- leia korduvkasutatavad objektid ja loo esialgne toores prototüüp;
- tee mõned testid, et leida vigu prototüübis;
- küsi kliendilt tagasisidet prototüübi kohta;

OO arendusmudel.Näide (2)

- muuda analüüsimudelit vastavalt sellele, mida õppisid prototüübist, kavandamisest ja kliendi tagasisidest;
- täpsusta kavandit arvestades muudatusi;
- kodeeri uued objektid, mis ei ole valmiskujul saadaval;
- pane kokku uus prototüüp, kasutades nii teegis olnud klasse kui uusi klasse;
- tee mõned testid, et leida vigu prototüübis;
- küsi kliendilt tagasisidet prototüübi kohta;
- Jne... (tekib tsükkel - iteratsioon).

Standardiseerimine

Standardi vajadus – **miks?**

Põhjus: Meetodite paljusus ja nende pooldajate omavaheline leppimatus. (*Q: What is the difference between a methodologist and a terrorist? A: You can negotiate with a terrorist.*)

- Booch'i ja Rumbaugh 1. mustand märts 1995 (*Unified Method*)
- 1996 Jacobson ühines nendega
- 1996 OMG üleskutse standardi loomiseks, B., R. ja J. vastasid UML-i versiooniga 1.0
- Märts 2003 – ver 1.5
- Oktoober 2004 - ver 2.0 üldstruktuur
- Praegune versioon UML 2.1.2 (jaanuar 2008)

UML

- **UML** (*Unified Modeling Language*) on graafiline modelleerimiskeel, mis aitab OO-tehnikas kirjeldada süsteemi elemente, mõisteid ja seoseid; kirjeldada, visualiseerida, kontrueerida ja dokumenteerida tarkvarasüsteemi detaile.
- UML on **OMG** (*Object Modeling Group*) poolt tunnustatud tehnoloogia novembrist 1997
- **Põhiautorid** *Grady Booch, Ivar Jacobson ja James Rumbaugh*, kolm sõpra (*three amigos*)

Protsess ei ole UML-i loojate eesmärk. **RUP** (*Rational Unified Process*) on B., R. ja J. poolt pakutud ühe võimaliku protsessina. UML ei ole otseselt seotud ühegi protsessiga.

UML-i omadused

- On lihtsalt õpitav ja semantiliselt (täenduslikult) rikas **visuaalse modelleerimise** keel.
- Kasutab üheselt arusaadavat mõistete hulka.
- Ühendab Booch'i, OMT ja Objectory mudelkeeled.
- Sisaldab ka teiste modelleerimiskeelte ideid.
- Koondab “tööstusharu” parimad kogemused.
- Arvestab kaasaegse tarkvaraarenduse teemadega: skaleeritavus, hajusus, paralleeltöötlus, ...
- On paindlik erinevates protsessides kasutamiseks.
- Lubab mudeleid omavahel seostada.

Tarkvarasüsteemi vaated (1)

Süsteemi saab esitada viie erineva UML-i vaate abil, mis kirjeldavad süsteemi erinevatest vaatenurkadest

- **Kasutajavaade:**
iseloomustab toodet kasutaja poolt nähtuna. Kasutusmallid. Lõppkasutajate tegevus süsteemiga.
- **Struktuurivaade:**
andmeid ja funktsionaalsust vaadatakse süsteemi seest, staatilised seosed (klassid, objektid, nende vahelised seosed)

Tarkvarasüsteemi vaated (2)

- Käitumisvaade:
süsteemi käitumine (suhtlus ja koostöö süsteemi struktuursete osade vahel)
- Realisatsioonivaade:
struktuuri ja käitumise aspekte esitatakse nii, nagu see realiseerida tuleb.
- Ümbruskonnavaade:
ümbruskonna struktuur ja käitumine, mille jaoks süsteem ehitatakse.

NB! Ühe mudeli piires ei tohi erinevad vaated segamini minna – nt kasutajavaatesse ei sobi realisatsiooni detailid.

Mudelid - üldine jaotus

- **Struktuurimudel** e staatiline mudel kirjeldab süsteemis olevate objektide struktuuri klasside, liideste, atribuutide ja seostega.
- **Käitumismudel** e dünaamiline mudel kirjeldab objektide käitumist süsteemis - meetodeid, suhtlemist, koostööd, olekute ajalugu.

Samal põhimõttel jaotatakse UML-i skeemid.

UMLi struktuuriskeemid

Struktuuriskeemid struktuurimudelite koostamiseks:

- Klassiskeem (*class diagram*)
- Komponentiskeem (*component diagram*)
- Liitstruktuuri skeem (*composite structure diagram*)
- Eviusskeem (*deployment diagram*)
- Objektiskeem (*object diagram*)
- Paketiskeem (*package diagram*)

UMLi käitumisskeemid

Käitumisskeemid käitumismudelite koostamiseks

- Tegevusskeem (*activity diagram*)
- Kasutusmalliskeemi (*use case diagram*)
- Olekumasinaskeem (*state machine diagram*)
- Interaktsiooniskeemid (*interaction diagram*)
 - Järgnevusskeem (*sequence diagram*)
 - Suhtlusskeem (*communication diagram*, oli *collaboration diagram*)
 - Interaktsiooni ülevaateskeem (*interaction overview d.*)
 - Ajadiagramm (*timing diagram*)

Kasutajavaade

Kasutajate nõudmised, kasutuslood,
kasutusmalliskeem, stsenaariumid.

Kasutajavaade ja kasutuslood

- **Kasutajate soovid/nõuded** – saame teada kasutajalt (intervjuud jne). Nad on aluseks süsteemi loomisel.
- Kes on kasutajad?
 - **Inimene** – interaktiivne süsteem inimese jaoks
 - **Masin** – programm juhib mingit aparati
 - **Teine programm** – programm juhib teiste programmide tööd.

NB! Kasutaja ei ole ainult inimene!!

- **Kasutajamudeli** (kasutusmudeli) ja **kasutusmalliskeemi** mõte lähtub I. Jacobson'i töödest ja tema *Objectory*-meetodist (1994).

Nõuded

- **Nõuded** (*requirements*) on süsteemi “oskused” ja tingimused, millele ta vastama peab.
- Kõige olulisem nõuete analüüsil on leida, selgeks rääkida, teistele edasi anda ja meelde jätta, mida tegelikult vaja on.
- Meelde jätmine eeldab kliendile ja arendusmeeskonna liikmele arusaadavat üleskirjutamist.
- UP raames algab nõuete leidmine algatusfaasis (kuid ei lõpe seal!)

FURPS+ (1)

UP jaoks pakutakse kasutuslugude süstematiseerimiseks **FURPS+** mudelit:

- **Funktsionaalsus** (*Functionality*) – võimalused, suutlikus, turvalisus
- **Kasutuskõlblikkus** (*Usability*) – inimfaktor, abi, dokumentatsioon
- **Töökindlus** (*Reliability*) – vigade tihedus, parandatavus
- **Suutvus** (*Performance*) – vastamise aeg, korrektsus, kättesaadavus, ressursside kasutamine
- **Toetatavus** (*Supportability*) – adapteeritavus, hooldatavus, rahvusvahelisus, konfigureeritavus

FURPS+ (2)

- **Implementation** – ressursside limiidid, keeled, vahendid
- **Interface** – piirangud liidestamisel väliste süsteemidega
- **Operations** – süsteemi haldamine
- **Legal** - litsentsid

Kategoriseerimise eesmärk on nõuete kergem ülesleidmine. Igasse kategooriasse peaks vähemalt mõned nõuded sattuma.

Nõuded UP tehistes

- Nõuded jaotatakse erinevate UP tehiste vahel:
 - **Kasutusloomudel** (*Use-Case Model*) - kirjeldab funktsionaalsed nõuded ja nendega seonduvad mittefunktsionaalsed nõuded.
 - **Täiendav spetsifikatsioon** (*Supplementary Specification*) - Kirjeldab teisi nõudeid.
 - **Ärireeglid** (*Business Rules*) – valdkonnas valitsevad reeglid, millega tuleb arvestada (mittefunktsionaalne).
 - jne

Tegija ehk roll

Kasutusmalli oluliseks osaks on **tegija** (*actor*) ehk **roll**.

- Tegija on süsteemiväline mõiste.
- **Aktiivne tegija** suhtleb süsteemiga, saates talle teateid ja algatades kasutusmalle stiimulite läbi.
- **Passiivne tegija** osaleb kasutusmallis.
- Tegija ei ole Kaarel Kaalikas vaid nt süs-admin, raamatupidaja, koristaja jne.
- Üks inimene, mitu rolli. Üks roll, mitu inimest.

Kasutusmall

Kasutusmall (*use case*) on toimingute jada tegija (kasutaja) ja arvutisüsteemi vahel.

Mall peab vastama järgmistele **tingimustele**:

- Kirjeldab funktsionaalseid nõudeid ja esitab kasutamise stsenaariumid, mis on kooskõlastatud kliendi ja arendaja vahel.
- Annab selge ja ühemõttelise kirjelduse sellest, kuidas lõppkasutaja ja uus tarkvarasüsteem omavahel suhtlevad.
- Annab aluse funktsionaalsuse testimiseks.

Kasutusmalli mudel

- **Kasutusmalli mudel** (*use case model*) on süsteemi vaade, mis kirjeldab süsteemi käitumist sellisena, nagu seda näeb kasutaja.
- Kasutusmalli mudel on üks soovituslikke UP tehiseid, mis annab edasi süsteemi funktsionaalsust.
- Kasutusmalli mudel sisaldab tegijaid, stsenaariumeid ja kasutusmalle.
- Kasutusmalli mudel on valdavalt tekstidokument, kuid võib sisaldada ka skeeme.

Kasutusmallide esitamine

Kasutusmalli esitamine: tekstina või pildina.

- Tekstina esitatakse näiteks tabeli vormis:
 - Nimi
 - Osalejad
 - Kirjeldus (kasutaja ja süsteemi tegevused)
 - Tüüp (võib eristada nt: nähtav, varjatud, täiendav, ... või siis tähtsuse järgi)
- Pildina esitamiseks on UML-i kasutusmalliskeem, kuid sellega ei tohi piirduda.

Kasutusmalliskeem

- UML-i üks käitumisskeemidest on **kasutusmalliskeem** (*use case diagram*)
- Skeemi abil saab luua üldise pildi sellest, kes süsteemi kasutavad ja millist funktsionaalsust vajavad.
- Kasutusmalliskeem annab väga üldise pildi ja ei sobi ainukeseks funktsionaalsuse esitamisevormiks.

Kasutusmalliskeemi osad

- **Tegija e roll** (*actor*) - kasutab süsteemi. Tegija ei pea olema inimene, vaid ka teine süsteem, mis antud süsteemi käest midagi tahab. Skeemil: kriipsujuku nimega.
- **Kasutusmall** (*use case*) - tegevused, mida tegija süsteemis teha saab. Skeemil: ellips, sees nimi (soovitavalt tegusõnaga).
- **Süsteemi piir** (*system boundary*) - piirjoon füüsilise süsteemi ja väliste tegijate vahel. Skeemil: ristkülik.
- **Link** (*link*) - seos tegija ja kasutusmalli vahel ning kasutusmallide vahel (vt järgmine slaid). Skeemil: pidev joon.

Erinevad seosed (1)

- **Side** (*association*) – tegija osalus kasutusmallis – mida tahab/saab teha. Skeemil: pidev joon.
- **Üldistus** (*generalization*) – seos spetsiifilisema ja üldisema kasutusmalli või tegija vahel. Skeemil: joon, suur seest tühi nool otsas.
- **Sisaldab** (*include*) – baaskasutusmalli käitumisse lisatakse teise kasutusmalli tegevused. Skeemil: katkendjoon lahtise noolega + <<*include*>>, suund lisatava malli poole
- **Laiendab** (*extends*) – laiendav kasutusmall täiendab baaskasutusmallis olevat käitumist. Skeemil: Katkendjoon lahtise noolega + <<*extends*>>, suund baasmalli poole

Erinevad seosed (2)

Kasuta seoseid järgmiselt:

- **'laiendab' seost** - kui on tegemist sama kasutusmalli erinevate variantidega
- **'sisaldab' seost** - kui erinevates kasutusmallides on osa tegevusi sarnased

Parem ära kasuta erinevaid seoseid üldse!

Stsenaarium

- **Stsenaarium** - sammud, mis kirjeldavad täpsemalt interaktsiooni kasutaja ja süsteemi vahel.
- Stsenaariumi üleskirjutamiseks sobib tabel ja tekst), tegevusskeem. Võimalik koostis:
 - Nimi
 - Eesmärk
 - Tase
 - Õnnestumise peastsenaarium (*main success scenario*)
 - Laiendused (õnnestunud ja läbikukkunud).

Stsenaarium. Näide (1)

Õnnestumise peastsenaarium

Nimi: Toote ostmine e-poes

Tase: peamine

- 1.Klient vaatab toodete kataloogi ja valib kaubad.
- 2.Klient registreerib ostu.
- 3.Klient täidab kohaletoimetamise info (aadress, tüüp – kiir/tavaline, jne).
- 4.Süsteem esitab hinnad, kaasaarvatud kohale toomine.
- 5.Klient täidab krediitkaardi info.
- 6.Süsteem autoriseerib ostutehingu.
- 7.Süsteem kinnitab müügi.
- 8.Süsteem saadab kliendile kinnitava e-maili.

Stsenaarium.Näide (2)

Laiendused:

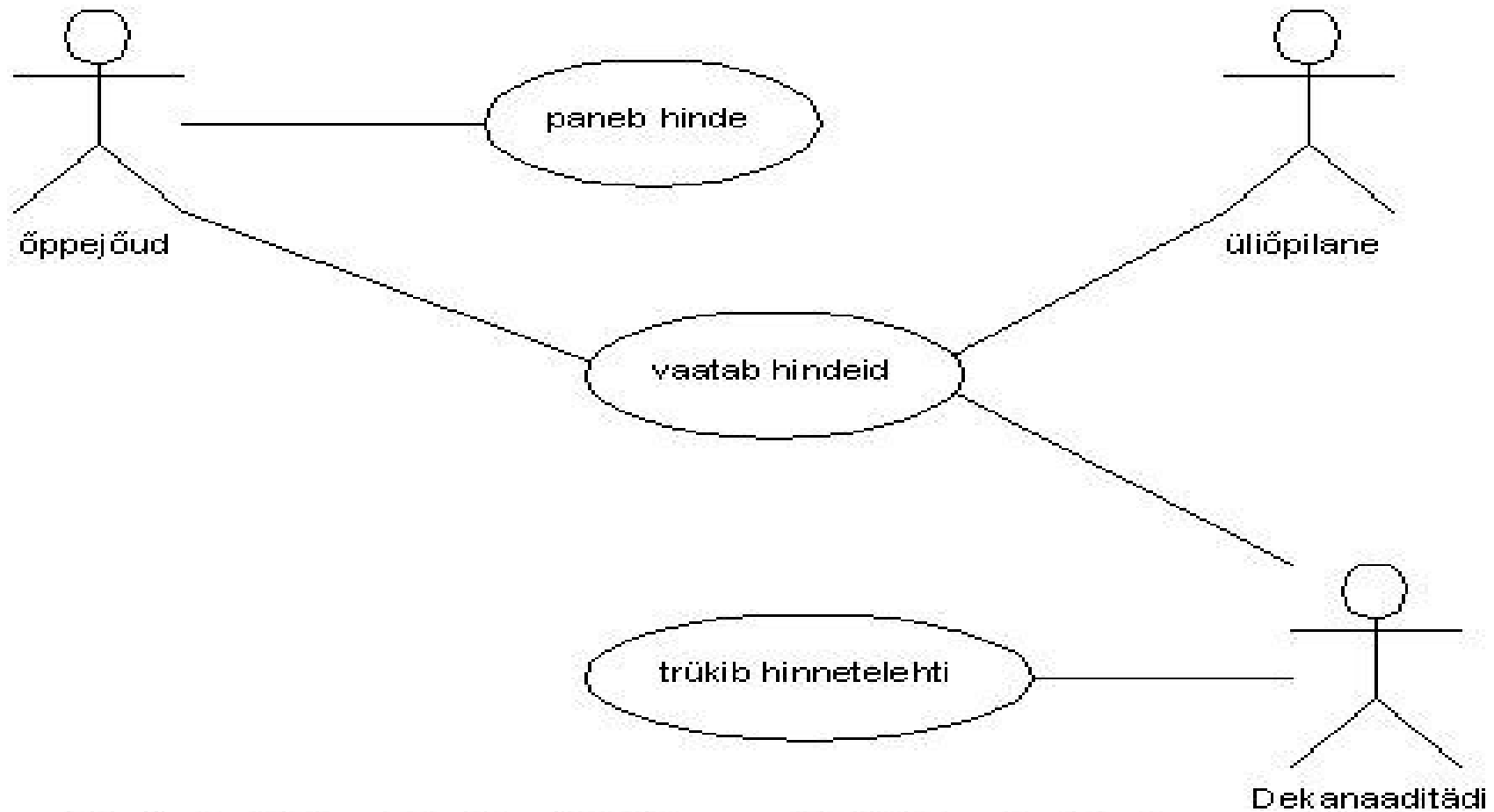
3a. Klient on püsiklient

- Süsteem pakub kliendi jaoks tavalise kohaletoimetamise jms info.
- Klient võib seda infot muuta. Tagasi põhistsenaariumisse

6a Süsteemil ei õnnestu krediitkaardi infot autoriseerida

- Klient võib uuesti sisestada krediitkaardi info või tegevuse lõpetada.

Kasutusmalliskeem



Created with Poseidon for UML Community Edition. Not for Commercial Use.

Millal ja milleks?

- Kasutusmallid aitavad mõista süsteemi **funktsionaalseid nõudeid**.
- Kasutusmall esitab süsteemi välist vaadet – mida tahab **kasutaja**.
- Keskendu stsenaariumide väljakirjutamisele. See on olulisem kui skeem.
- Skeemidele ära kuluta liialt aega ja ära tee neid keerulisteks ning detailseteks.
- Sobib suhtlemiseks “tavaliste” inimestega.
- Võib aidata kasutajaliidese loomisel, testimise ettevalmistamisel.