Ohjelmistotekniikan menetelmät

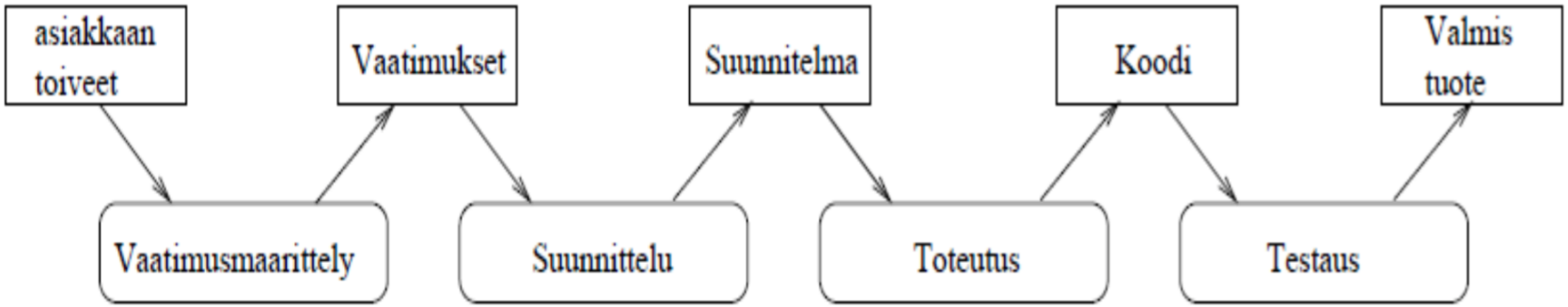
luento 2, 8.11.2016

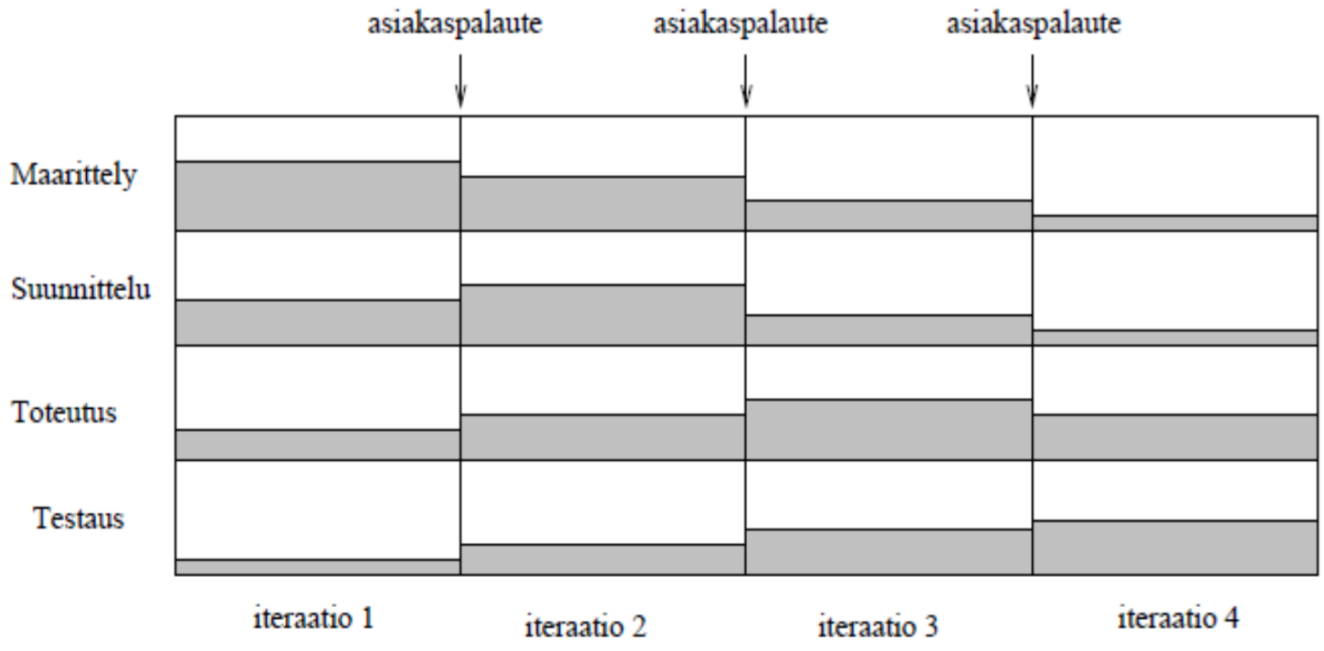
Kertaus - ohjelmistotuotantoprosessi

Käytetystä menetelmästä riippumatta
ohjelmistotuotantoprosessissa tapahtuu seuraavia aktiviteettejä

1. **Vaatimusmäärittely**
   - Mitä halutaan?
2. **Suunnittelu**
   - Miten tehdään?
3. **Toteutus**
   - Ohjelmoidaan
4. **Testaus**
   - Varmistetaan että toimii niin kuin halutaan
   - Yksikkötestaus, integraatiotestaus, järjestelmä/hyväksymätestaus, regressiotestaus
5. **Ylläpito**
   - Korjataan bugeja ja laajennetaan ohjelmistoa

Kertaus - ketterä vs. vesiputous





Mallintaminen

- Ohjelmistotuotantoon liittyy **mallintaminen**, eli kyky tuottaa erilaisia kuvauksia, joita tarvitaan ohjelmiston kehittämisen yhteydessä
- Mallit toimivat kommunikoinnin välineinä
- *Mitä ollaan tekemässä, miten ollaan tekemässä, mitä tehtiin?*
- Kurssi oli aiemmin nimeltään *Ohjelmistojen mallintaminen*, painopiste on hieman muuttunut, mutta mallintaminen on edelleen vahvasti mukana

Mallintaminen

Malli on abstrakti kuvaus mielenkiinnon alla olevasta kohteesta

- Perinteiset insinöörialat perustuvat malleihin
  - Esim. siltaa rakentaessa tarkat lujuuslaskelmat (=malli)
  - Näihin perustuen tehdään piirustukset, eli malli siitä miten silta pitää toteuttaa (=edellistä hieman tarkempi malli)
- Kuvaa vain olennaisen

Mallintaminen

Malli on abstrakti kuvaus mielenkiinnon alla olevasta kohteesta

- Perinteiset insinöörialat perustuvat malleihin
  - Esim. siltaa rakentaessa tarkat lujuuslaskelmat (=malli)
  - Näihin perustuen tehdään piirustukset, eli malli siitä miten silta pitää toteuttaa (=edellistä hieman tarkempi malli)
- Kuvaa vain olennaisen
- Käyttötarkoitusta varten liian tarkat tai liian ylimalkaiset mallit epäoptimaalisia
- Mitä on olennaista, riippuu mallin käyttötarkoituksesta
  - Metron linjakartta on hyvä malli julkisen liikenteen käyttäjälle
  - Autoilija taas tarvitsee tarkemman mallin eli tiekartan
  - Helsingin keskustassa kävelija taas tarvitsee tarkempaa kartta

Mallintaminen

Mallin näkökulma ja abstraktiotaso

- Mallien **abstraktiotaso** vaihtelee:
  - Abstraktimpi malli käyttää korkeamman tason käsitteitä
  - Konkreettisempi malli taas on yksityiskohtaisempi ja käyttää "matalamman"tason käsitteitä ja kuvaa kohdetta tarkemmin

Mallintaminen

Mallin näkökulma ja abstraktiotaso

- Mallien **abstraktiotaso** vaihtelee:
  - Abstraktimpi malli käyttää korkeamman tason käsitteitä
  - Konkreettisempi malli taas on yksityiskohtaisempi ja käyttää "matalamman"tason käsitteitä ja kuvaa kohdetta tarkemmin
- Mallien **näkökulma** vaihtelee:
  - Jos kaikki yritetään mahduttaa samaan malliin, ei lopputulos ole selkeä
  - Malli kuvaa usein korostetusti tiettyä näkökulmaa
  - Eri näkökulmat yhdistämällä saadaan idea kokonaisuudesta

Oikea malli oikeaan tarkoitukseen!

Mallintaminen

Mallin näkökulma ja abstraktiotaso: asunnon malli

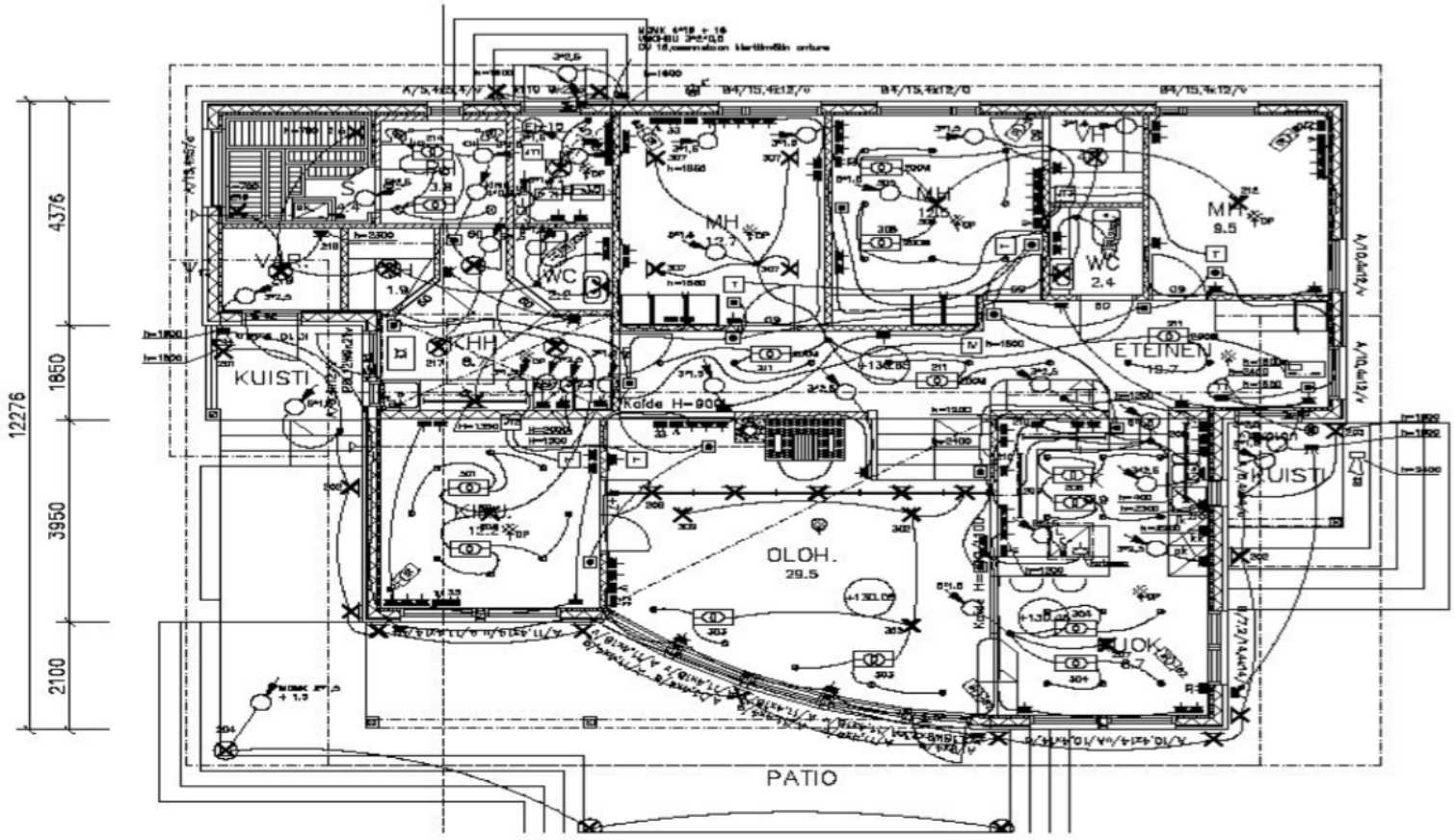
- Hyvin abstrakti kuvaus asunnosta:
  - 78 m^2, 3h, keittiö ja sauna

Mallintaminen

Mallin näkökulma ja abstraktiotaso: asunnon malli

- Hyvin abstrakti kuvaus asunnosta:
  - 78 m^2, 3h, keittiö ja sauna
- Hieman konkreettisempi

- Vielä konkreettisempi malli
  - näkökulmana sähkösuunnitelma



- Sähkösuunnitelma on oleellinen mallinnusnäkökulma sähköasentajan kannalta
- asunnon ostajalle kyseessä on kuitenkin todennäköisesti liian tarkasti väärään näkökulmaan keskittyvä malli

Mallintaminen

Ohjelmistojen mallintaminen

Entä ohjelmistojen mallintaminen?

- Vaatimusdokumentissa mallinnetaan mitä järjestelmän toiminnallisuudelta halutaan
- Suunnitteludokumentissa mallinnetaan...
  - Järjestelmän arkkitehtuuri eli jakautuminen tarkempiin komponentteihin
  - Yksittäisten komponenttien toiminta
- Toteuttaja käyttää näitä malleja ja luo konkreettisen tuotteen
- Testauksessa verrataan lopputuotetta vaatimusdokumentin malliin
- Vaatimuksien mallit yleensä korkeammalla abstraktiotasolla kuin suunnitelman mallit
  - Vaatimus ei puhu ohjelman sisäisestä rakenteesta toisin kuin suunnitelma

Mallintaminen

Ohjelmistojen mallintaminen

- Myös ohjelmistojen malleilla on erilaisia näkökulmia
- Jotkut mallit kuvaavat rakennetta...
  - Mitä komponentteja järjestelmässä on?
- Jotkut taas toimintaa...
  - Miten komponentit kommunikoivat?

Eri näkökulmat yhdistämällä saadaan idea kokonaisuudesta

Mallintaminen

Mallinnuksen kaksi suuntaa

- Usein mallit toimivat apuna kun ollaan rakentamassa jotain uutta, eli
  - Ensin tehdään malli, sitten rakennetaan esim. silta
- Toisaalta esim. fyysikot tutkivat erilaisia fyysisen maailman ilmiöitä rakentamalla niistä malleja ymmärryksen helpottamiseksi
  - Ensin on olemassa jotain todellista josta sitten luodaan malli
- Ohjelmistojen mallinnuksessa myös olemassa nämä **kaksi mallinnussuuntaa**
  - Ohje toteuttamiselle: malli $\rightarrow$ ohjelma
  - Apuna asiakkaan ymmärtämiseen: ohjelma $\rightarrow$ malli
    - ns. takaisinmallinnus

Ohjelmistojen mallinnuskäytännöt

Oliomallinnus ja UML

Ohjelmistojen mallinnuskäytännöt

Oliomallinnus ja UML

Miten mallintaa ohjelmistoja?

- Pitkään tilanne oli sekava ja on sitä osin edelleen
- Suosituimmaksi tavaksi on noussut **oliomallinnus**
- Unified Modelling Language eli UML
- Oliomallinnusta varten kehitetty standardoitu kuvaustekniikka
  - 1990 luvulta, nykyinen versio 2.5 (vuonna 2015)
- UML:ssä nykyään 13 erityyppistä kaaviota
  - Eri näkökulmille omat kaavionsa
- UML standardi ei määrittele miten ja missä tilanteissa kaavioita tulisi käyttää
  - Tätä varten olemassa useita oliomenetelmiä

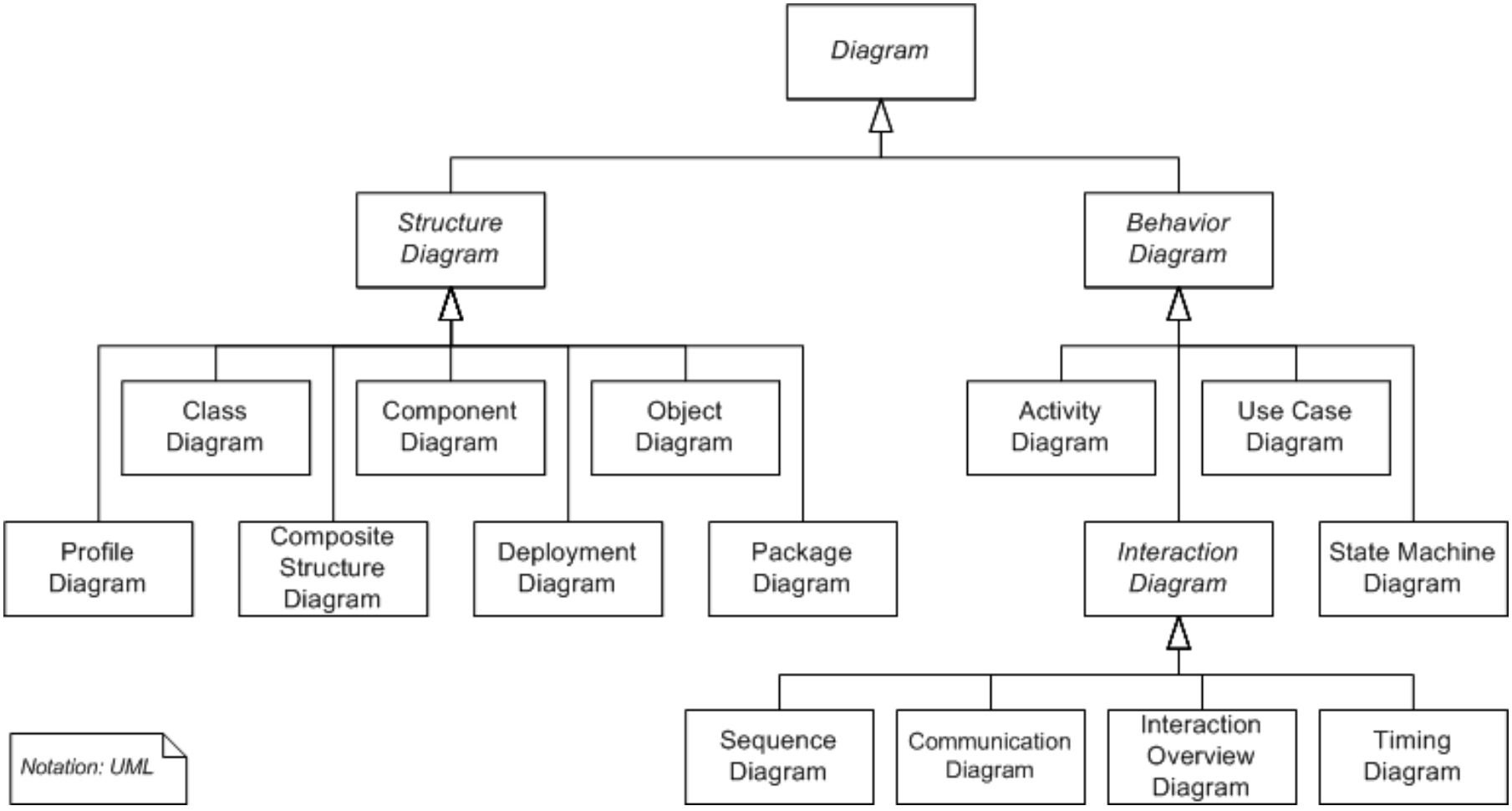


Diagram
Structure Diagram
Behavior Diagram
Class Diagram
Component Diagram
Object Diagram
Activity Diagram
Use Case Diagram
Profile Diagram
Composite Structure Diagram
Deployment Diagram
Package Diagram
Interaction Diagram
State Machine Diagram
Sequence Diagram
Communication Diagram
Interaction Overview Diagram
Timing Diagram
Notation: UML

Ohjelmistojen mallinnuskäytännöt

UML:n käyttötapa

- UML-standardi määrittelee kaavioiden syntaksin eli oikeaoppisen piirtotavan suhteellisen tarkasti
  - Eri versioiden välillä pieniä muutoksia
  - Vanhojen standardien mukaisia kaavioita näkyy yhä
- Jotkut suosivat UML:n käyttöä tarkasti syntaksia noudattaen
  - Kaaviot piirretään tällöin usein tietokoneavusteisella suunnitteluvälineellä
  - Nykyään tämä suuntaus alkaa olla jo melko harvinaista
- On myös UML:n luonnosmaisemman käytön puolestapuhujia
  - ns. ketterä mallinnus
  - Kaaviot ennenkaikkia kommunikoinnin apuväline
  - Tärkeimmät kuvat (ehkä) siirretään sähköiseen muotoon
    - Digikuva tai uudelleenpiirto editorilla

Käyttötapausmalli

Käyttötapausmalli

- Kuten muistamme, ohjelmistotuotatoprosessin ensimmäinen vaihe on *vaatimusmäärittely*
- Vaatimuksen jakautuvat kahteen luokkaan
- **Toiminnalliset vaatimukset**
  - Mitä toimintoja ohjelmassa on?
  - Esim. kurssihallintaohjelmistossa:
    - *Opetushallinto voi syöttää kurssin tiedot järjestelmään*
    - *Opiskelija voi ilmoittautua valitsemalleen kurssille*
    - *Opettaja voi syöttää opiskelijan suoritustiedot*
- **Ei-toiminnalliset vaatimukset** (eli ympäristön rajoitteet)
  - Toteutusympäristö, suorituskykyvaatimukset, ...
- **Vaatimusmäärittely** ei yleensä ota kantaa ohjelman sisäisiin teknisiin ratkaisuihin, ainoastaan siihen miten ohjelmiston toiminnallisuudet näkyvät käyttäjälle

Käyttötapausmalli

- Nyt esiteltävä **käyttötapausmalli** (engl. use case model) on yksi tapa toiminnallisten vaatimusten ilmaisemiselle
  - Ei-toiminnallisten vaatimusten ilmaisemiseen käyttötapausmalli ei juuri ota kantaa, vaan ne on ilmaistava muuten
  - On olemassa muitakin tapoja toiminnallisten vaatimusten ilmaisuun esim. kurssilla Ohjelmistotuotanto esiteltävät *User storyt* eli käyttäjätarinat

Käyttötapausmalli

Käyttäjien tunnistaminen

Hyvä tapa aloittaa vaatimusmäärittely on tunnistaa/etsiä rakennettavan järjestelmän käyttäjät.

- Kysymyksiä jotka auttavat:
  - Kuka käyttää järjestelmää?
  - Mihin muihin järjestelmiin kehitettävä järjestelmä on yhteydessä?
  - Kuka/mikä saa tietoa järjestelmästä?
  - Kuka/mikä toimittaa tietoa järjestelmään?
- Käyttäjä on oikeastaan **rooli**
  - Missä roolissa toimitaan järjestelmän suhteen
  - Yksi ihminen voi toimia monessa roolissa ...

Käyttötapausmalli

Käyttäjien tunnistaminen

TKTL:n kurssi-ilmoittautumisjärjestelmä

- Käyttäjärooleja
  - Opiskelija
  - Opettaja
  - Opetushallinto
  - Suunnittelija
  - Laitosneuvosto
  - Tilahallintojärjestelmä
  - Henkilöstöhallintajärjestelmä

Osa käyttäjistä yhteydessä järjestelmään vain epäsuorasti.

- Osa "*käyttäjistä*"on muita järjestelmiä
  - Sana käyttäjä ei ole terminä tässä tilanteessa paras mahdollinen
  - Englanninkielinen termi **actor** onkin hieman suomenkielistä termiä kuvaavampi

Käyttötapausmalli

Käyttötapaus

- **Käyttötapaus** (engl. use case) kuvaa käyttäjän ohjelman avulla suorittaman tehtävän.
  - *Miten käyttäjä kommunikoi järjestelmän kanssa tietyssä käyttötilanteessa?*
- Esimerkiksi Kurssi-ilmoittautumisjärjestelmällä käyttötapaus *Opiskelijan ilmoittautuminen*
  - Mitä vuorovaikutusta käyttäjän ja järjestelmän välillä tapahtuu kun opiskelija ilmoittautuu kurssille?
- Yksi käyttötapaus on looginen, "isompi" kokonaisuus
  - Käyttötapauksella lähtökohta
  - Ja merkityksen omaava lopputulos
  - Eli pienet asiat, kuten "syötä salasana" eivät ole käyttötapauksia
    - Kyseessä pikemminkin yksittäinen operaatio, joka voi sisältyä käyttötapaukseen

Käyttötapausmalli

Käyttötapauksen kuvaaminen

- **Kuvataan tekstinä**
- Ei ole olemassa täysin vakiintunutta tapaa kuvaukseen (esim. UML ei ota asiaan kantaa)
- Kuvauksessa mukana usein tietyt osat:
  - Käyttötapauksen nimi
  - Käyttäjät
  - Laukaisija
  - Esiehto
  - Jälkiehto
  - Käyttötapauksen kulku
  - Poikkeuksellinen toiminta

Käyttötapaus: Opiskelija ilmoittautuu kurssille

- *Käyttäjä:* opiskelija
- *Tavoite:* saada kurssipaikka
- *Laukaisija:* opiskelijan tarve
- *Esiehto:* opiskelija on ilmoittautunut kuluvalla lukukaudella läsnäolevaksi
- *Jälkiehto:* opiskelija on lisätty haluamansa ryhmän ilmoittautujien listalle
- *Käyttötapauksen kulku:*

  1. Opiskelija aloittaa kurssi-ilmoittautumistoiminnon
  2. Järjestelmä näyttää kurssitarjonnan
  3. Opiskelija tutkii kurssitarjontaa
  4. Opiskeljia valitsee ohjelmiston esittämästä tarjonnasta kurssin ja ryhmän
  5. Järjestelmä pyytää opiskelijaa tunnistautumaan
  6. Opiskelija tunnistautuu ja aktivoi ilmoittautumistoiminnon
  7. Järjestelmä ilmoittaa opiskelijalle ilmoittautumisen onnistumisesta.

- *Poikkeuksellinen toiminta:*

  4a. Opiskelija ei voi valita ryhmää, joka on täynnä

  6a. Opiskelija ei voi ilmoittautua, jos hänelle on kirjattu osallistumiseste.

Käyttötapaus: Opiskelija ilmoittautuu kurssille

- **Esiehto**
  - Asioiden tila jonka on vallittava, jotta käyttötapaus pystyy käynnistymään
- **Jälkiehto**
  - Kuvaa mikä on tilanne käyttötapauksen onnistuneen suorituksen jälkeen
- **Laukaisija**
  - Mikä aiheuttaa käyttötapauksen käynnistymisen
  - Voi olla myös ajan kuluminen
  - Usein niin itsestään selvä ettei kannata merkata
- **Käyttötapauksen kulku**
  - Kuvaa onnistuneen suorituksen, usein edellisen sivun tapaan käyttäjän ja koneen välisenä dialogina
- **Poikkeuksellinen toiminta**
  - Mitä tapahtuu jos tapahtumat eivät etene onnistuneen suorituksen kuvauksen mukaan
  - Viittaa onnistuneen suorituksen dialogin numeroihin, esim. jos kohdassa 4 voi tapahtua poikkeus normaaliin kulkuun, kuvataan se askeleena 4a

Käyttötapaus: opiskelija peruu ilmoittautumisen

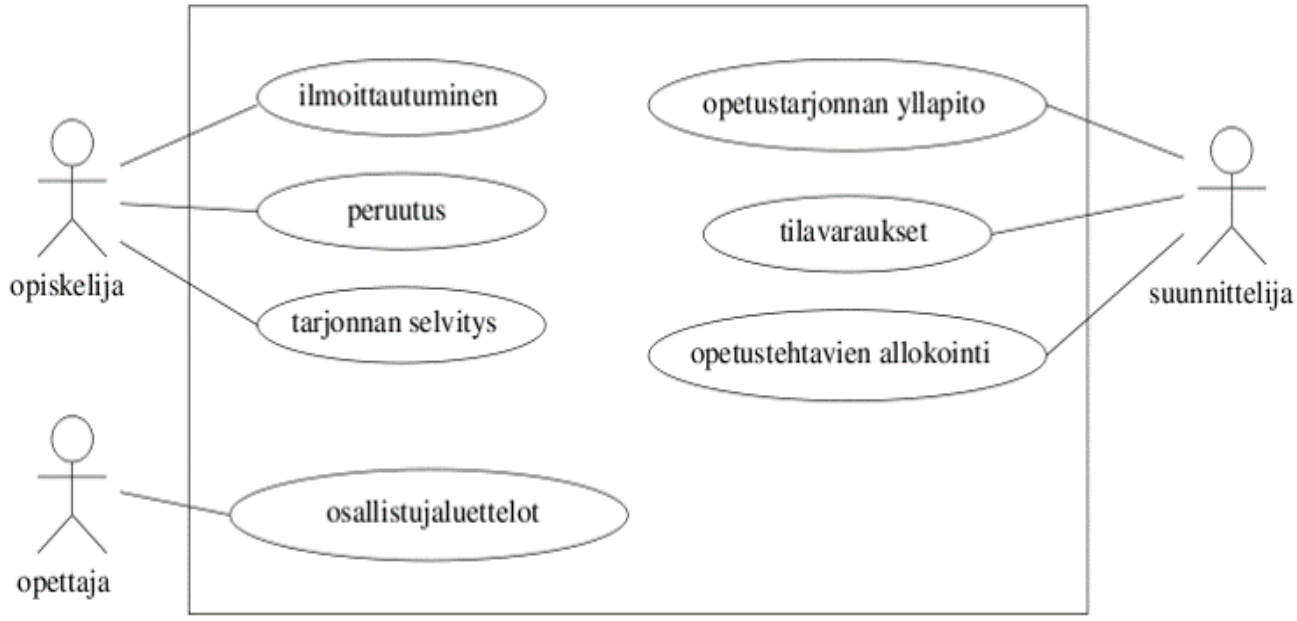
- *Käyttäjä:* opiskelija
- *Tavoite:* perua ilmoittautuminen, välttää sanktiot
- *Laukaisija:* opiskelijan tarve poistaa ilmoittautuminen
- *Esiehto:* opiskelija on ilmoittautunut tietylle kurssille
- *Jälkiehto:* opiskelijan ilmoittautuminen kurssille on poistettu
- *Käyttötapauksen kulku:*

  1. Opiskelija valitsee toiminnon "omat ilmoittautumiset"
  2. Järjestelmä pyytää opiskelijaa tunnistautumaan
  3. Opiskelija tunnistautuu
  4. Järjestelmä näyttää opiskelijan ilmoittautumiset
  5. Opiskeljia valitsee tietyn ilmoittatumisensa ja peruu sen
  6. Järjestelmä ilmoittaa opiskelijalle ilmoittautumisen peruuntumisesta

Käyttötapausmalli

Käyttötapauskaavio

- UML:n *käyttötapauskaavion* avulla voidaan kuvata käyttötapausten ja käyttäjien (engl. actor) keskinäisiä suhteita
- Kurssi-ilmoittautumisjärjestelmän “korkean tason” käyttötapauskaavio:



Käyttötapausmalli

Käyttötapauskaavio

- Käyttäjät kuvataan tikku-ukkoina
  - Olemassa myös vaihtoehtoinen symboli, joka esitellään pian
- Käyttötapaukset taas kuvataan järjestelmää kuvaavan neliön sisällä olevina ellipseinä
  - Ellipsin sisällä käyttötapauksen nimi
- Käyttötapausellipsiin yhdistetään viivalla kaikki sen käyttäjät
  - Kuvaan ei siis piirretä nuolia!

HUOM: Käyttötapauskaaviossa ei kuvata mitään järjestelmän sisäisestä rakenteesta

- Esim. vaikka tiedettäisiin että järjestelmä sisältää tietokannan, ei sitä tule kuvata käyttötapausmallissa

Käyttötapausmalli

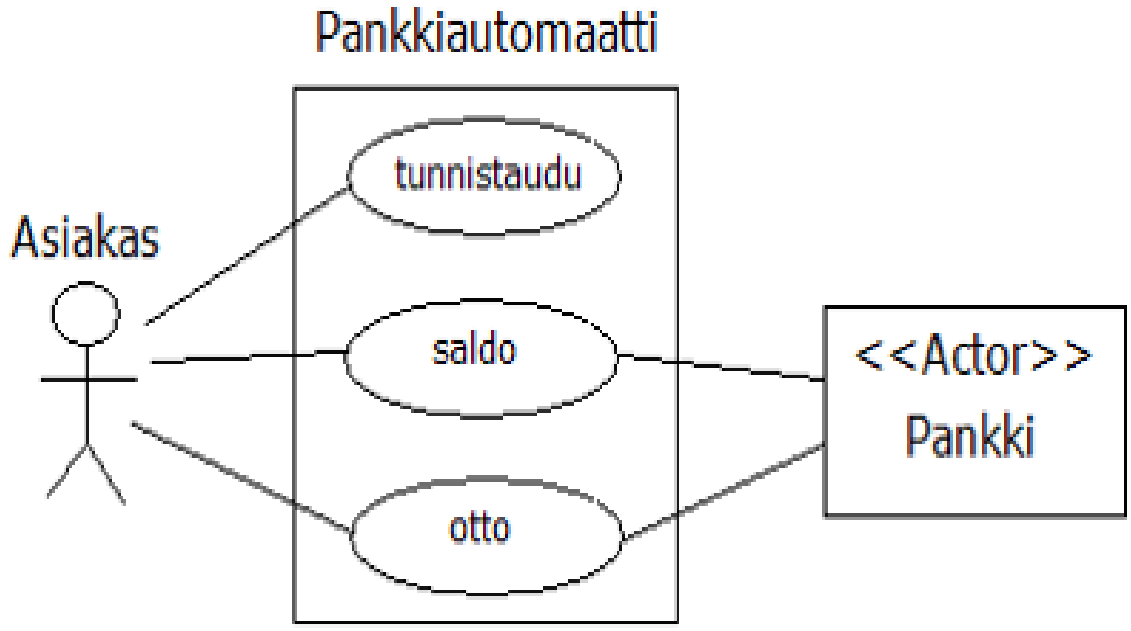
Käyttötapauskaavion käyttö

- Kaaviossa siis käyttötapauksista ainoastaan nimi
  - Käyttötapauksen sisältö kuvataan aina tekstuaalisena esityksenä
- Kaavio tarjoaa hyvän yleiskuvan järjestelmän käyttäjistä ja palveluista
  - Määrittelydokumentin alussa kannattaakin olla käyttötapauskaavio "sisällysluettelona"
- Jokainen käyttötapaus tulee kirjata tekstuaalisesti tarvittavalla tarkkuudella
  - Ei siis ole olemassa standardoitua tapaa käyttötapauksen kirjaamiseen
  - Ohjelmistoprojektissa tulee kuitenkin määritellä käyttötapauspohja, eli sopia joku yhteinen muoto, jota kaikkien käyttötapausten dokumentoinnissa noudatetaan

Käyttötapausmalli

Toinen esimerkki: pankkiautomaatin käyttötapaukset

- Käyttötapaukset ovat *tunnistaudu, saldo ja otto*
- Käyttötapausten käyttäjät eli toimintaan osallistuvat tahot ovat *Asiakas ja Pankki*
  - Alla on esitelty tikku-ukolle vaihtoehtoinen tapa merkitä käyttäjä eli laatikko, jossa merkintä <<actor>>



Käyttötapaus 1: otto

Tavoite Asiakas nostaa tililtään haluamansa määrän rahaa

Käyttäjät Asiakas, Pankki

Esiehto Kortti syötetty ja asiakas tunnistautunut

Jälkiehto käyttäjä saa tililtään haluamansa määrän rahaa.
Jos saldo ei riitä, tiliä ei veloiteta

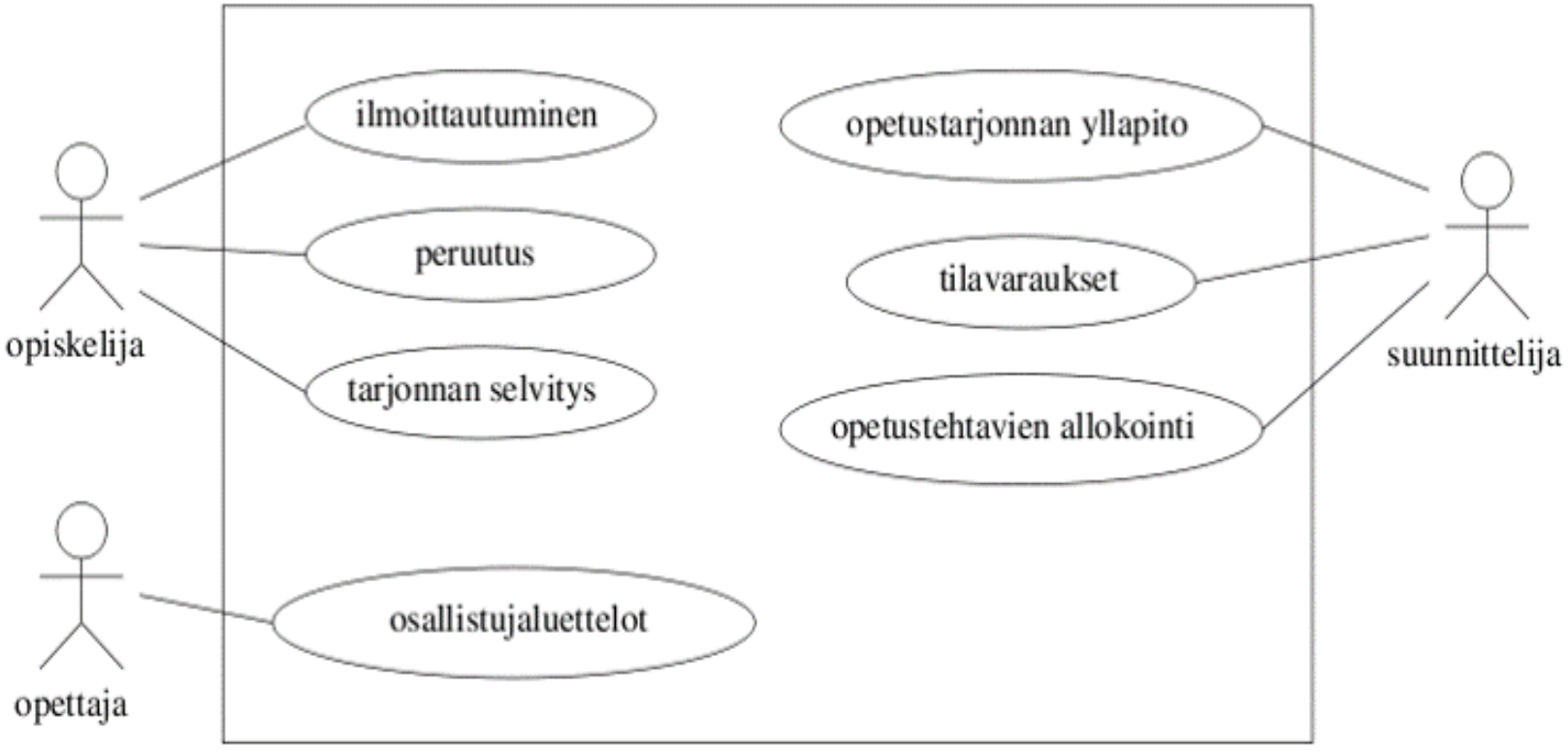
Käyttötapauksen kulku:

1. asiakas valitsee otto-toiminnon
2. automaatti kysyy nostettavaa summaa
3. asiakas syöttää haluamansa summan
4. pankilta tarkistetaan riittääkö asiakkaan saldo
5. summa veloitetaan asiakkaan tililtä
6. kuitti tulostetaan ja annetaan asiakkaalle
7. rahat annetaan asiakkaalle
8. pankkikortti palautetaan asiakkaalle

Käyttötapauksen kulku:

4a asiakkaan tilillä ei tarpeeksi rahaa, palautetaan kortti

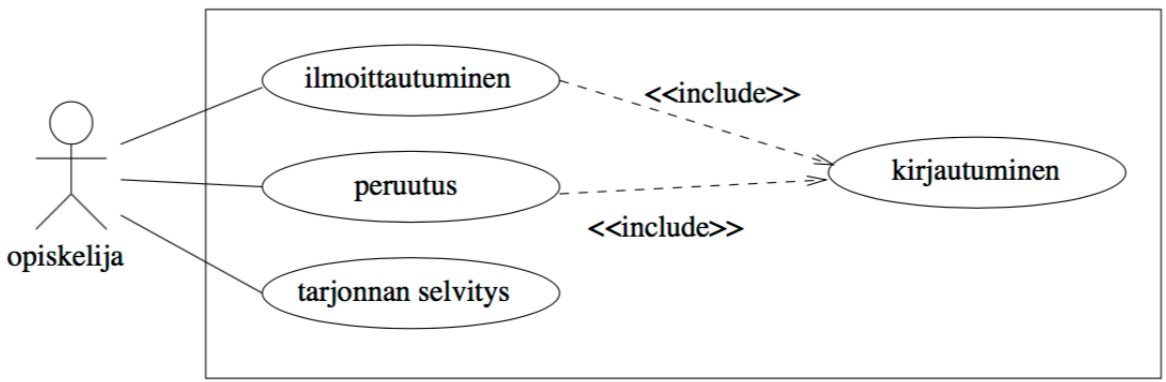
Palataan takaisin kurssi-ilmoittautumisjärjestelmään . . .



Käyttötapausmalli

Yhteiset osat

- Moneen käyttötapaukseen saattaa liittyä yhteinen osa
- Yhteisestä osasta voidaan tehdä "alikäyttötapaus", joka sisällytetään (include) pääkäyttötapaukseen
- Käyttötapauskaaviossa tätä varten merkintä <<include>>
  - katkoviivanuoli pääkäyttötapauksesta apukäyttötapaukseen
- Esim. käyttötapaus kirjautuminen suoritetaan aina kun tehdään ilmoittautuminen tai peruutus



- Sisällytetyn käyttötapauksen suorittaminen kannattaa merkitä käyttötapauksen tekstuaaliseen kuvaukseen, ks. seuraava kalvo

Käyttötapaus: opiskelija ilmoittautuu kurssille

- *Käyttäjä:* opiskelija
- *Tavoite:* saada kurssipaikka
- *Laukaisija:* opiskelijan tarve
- *Esiehto:* opiskelija on ilmoittautunut kuluvalla lukukaudella läsnäolevaksi
- *Jälkiehto:* opiskelija on lisätty haluamansa ryhmän ilmoittautujien listalle
- *Käyttötapauksen kulku:*

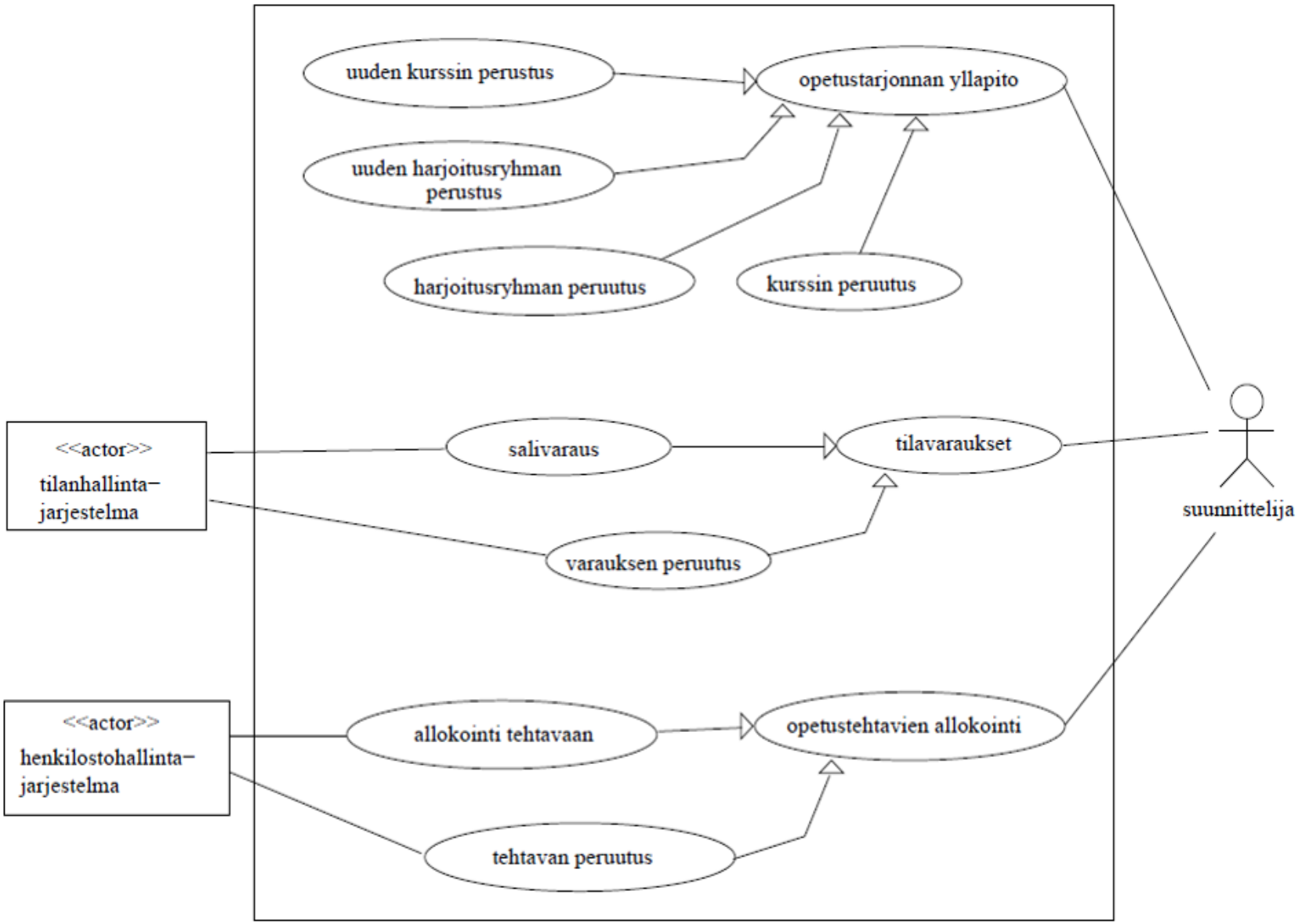
  1. Opiskelija aloittaa kurssi-ilmoittautumistoiminnon
  2. Järjestelmä näyttää kurssitarjonnan
  3. Opiskelija tutkii kurssitarjontaa
  4. Opiskeljia valitsee ohjelmiston esittämästä tarjonnasta kurssin ja ryhmän
  5. **Suoritetaan käyttötapaus kirjautuminen**
  6. Järjestelmä ilmoittaa opiskelijalle ilmoittautumisen onnistumisesta.

- *Poikkeuksellinen toiminta:*

Yleistetty ja erikoistettu käyttötapaus

- Suunnittelijan käyttötapauksista erityisesi opetustarjonnan ylläpito on hyvin laaja tehtäväkokonaisuus
- Voidaankin ajatella, että kyseessä on *yleistetty käyttötapaus*, joka oikeasti pitääkin sisällään useita konkreettisia käyttötapauksia, kuten
- Esim. voidaan ajatella, että *uuden kurssin perustus* ja *kurssin peruutus* ovat osa yleisettyä käyttötapausta, jota kutsumme nimellä *opetustarjonnan ylläpito*
- Seuraavalla sivulla esimerkki yleisen käyttötapauksen erikoistavat käyttötapaukset merkitään käyttötapauskaavioon

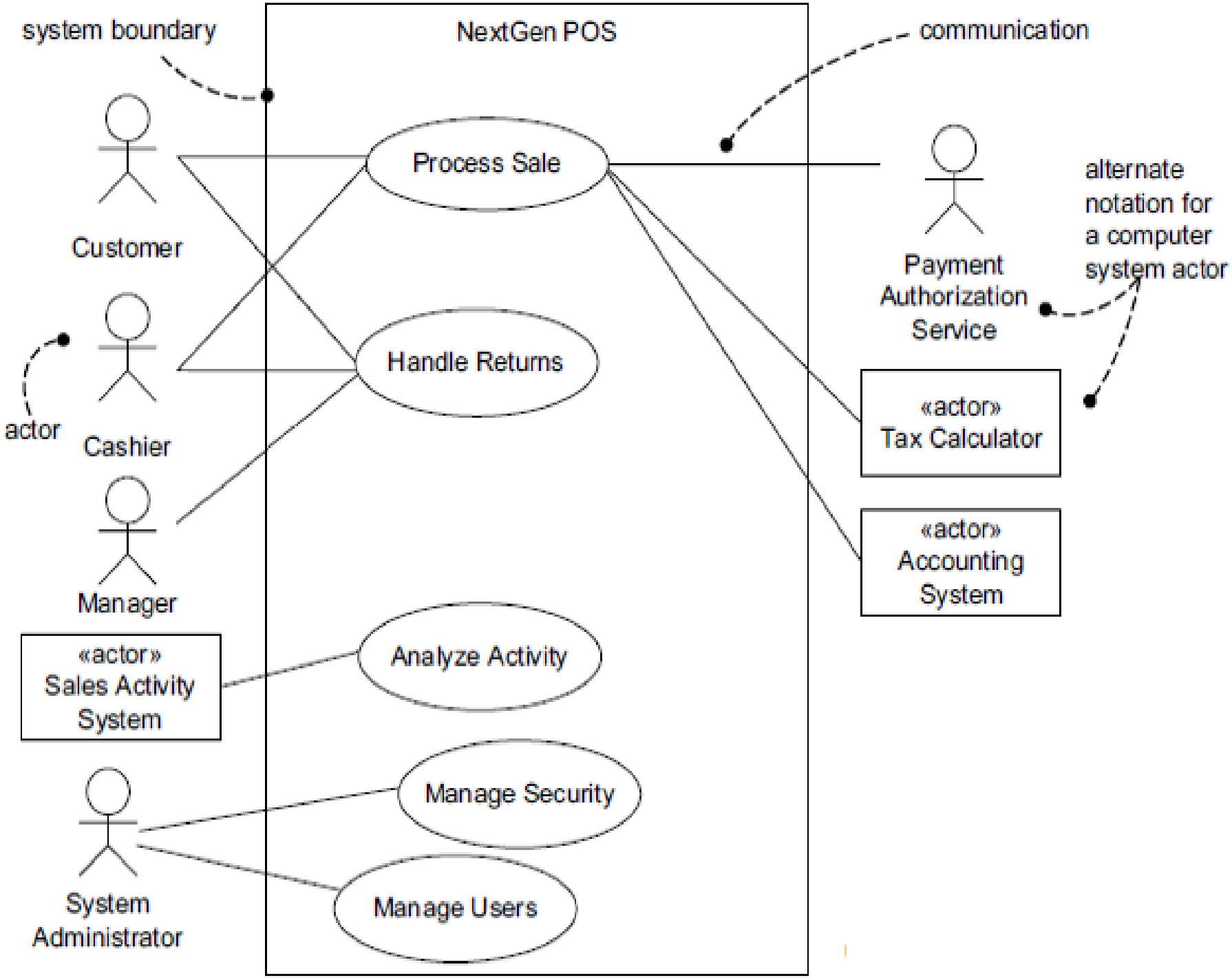
Yleistetty ja erikoistettu käyttötapaus



Realistisempi esimerkki: kassapäätejärjestelmä

- Craig Larmanin kirjasta *Applying UML and Patterns* [1]
- Aluksi etsitään järjestelmän käyttäjät
- Mietitään käyttäjien tavoitteita: mitä käyttäjä haluaa saada järjestelmällä tehtyä
- Käyttäjän tavoitteellisista toiminnoista (esim. käsittele ostos) tulee tyypillisesti käyttötapauksia
- Samalla saatetaan löytää uusia käyttäjiä (erityisesti ulkoisia järjestelmiä joihin järjestelmä yhteydessä)
- Hahmotellaan alustava käyttötapauskaavio

[1] Kirjan käyttötapausluku löytyy verkosta:
`http://www.craiglarman.com/wiki/index.php?title=Articles`


system boundary
NextGen POS
communication
Process Sale
Customer
Payment Authorization Service
alternate notation for a computer system actor
Handle Returns
actor
Cashier
«actor» Tax Calculator
«actor» Accounting System
Manager
«actor» Sales Activity System
Analyze Activity
Manage Security
System Administrator
Manage Users

Realistisempi esimerkki: kassapäätejärjestelmä

- Otetaan aluksi tarkasteluun järjestelmän toiminnan kannalta kriittisimmät käyttötapaukset
- Ensin kannattanee tehdä vapaamuotoinen kuvaus käyttötapauksista ("brief use case")

> **Process Sale:** A customer arrives at a checkout with items to purchase. The cashier uses the POS system to record each purchased item. The system presents a running total and line-item details. The customer enters payment information, which the system validates and records. The system updates inventory. The customer receives a receipt from the system and then leaves with the items.

Kuva: POS = point of sales terminal eli kassapääte

- Tarkempi käyttötapaus kirjoitetaan projektin sopiman käyttötapauspohjan määräämässä muodossa

Use Case UC1: Process Sale

Primary Actor: Cashier
Preconditions: Cashier is identified and authenticated.
Success Guarantee (Postconditions): Sale is saved. Tax is correctly calculated. Accounting and Inventory are updated. Commissions recorded. Receipt is generated. Payment authorization approvals are recorded.

Main Success Scenario (or Basic Flow):

1. Customer arrives at POS checkout with goods and/or services to purchase.
2. Cashier starts a new sale.
3. Cashier enters item identifier.
4. System records sale line item and presents item description, price, and running total. Price calculated from a set of price rules.

Cashier repeats steps 3-4 until indicates done.

5. System presents total with taxes calculated.
6. Cashier tells Customer the total, and asks for payment.
7. Customer pays and System handles payment.
8. System logs completed sale and sends sale and payment information to the external Accounting system (for accounting and commissions) and Inventory system (to update inventory).
9. System presents receipt.
10. Customer leaves with receipt and goods (if any).

Extensions (or Alternative Flows):

*a. At any time, System fails:

To support recovery and correct accounting, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Cashier restarts System, logs in, and requests recovery of prior state.
2. System reconstructs prior state.
   2a. System detects anomalies preventing recovery:
      1. System signals error to the Cashier, records the error, and enters a clean state.
      2. Cashier starts a new sale.

3a. Invalid identifier:

1. System signals error and rejects entry.

3b. There are multiple of same item category and tracking unique item identity not important (e.g., 5 packages of veggie-burgers):

1. Cashier can enter item category identifier and the quantity.

3-6a: Customer asks Cashier to remove an item from the purchase:

1. Cashier enters item identifier for removal from sale.
2. System displays updated running total.

3-6b. Customer tells Cashier to cancel sale:

1. Cashier cancels sale on System.

3-6c. Cashier suspends the sale:

1. System records sale so that it is available for retrieval on any POS terminal.

7a. Paying by cash:

1. Cashier enters the cash amount tendered.
2. System presents the balance due, and releases the cash drawer.
3. Cashier deposits cash tendered and returns balance in cash to Customer.
4. System records the cash payment.

7b. Paying by credit:

1. Customer enters their credit account information.
2. System sends payment authorization request to an external Payment Authorization Service System, and requests payment approval.
   2a. System detects failure to collaborate with external system:
      1. System signals error to Cashier.
      2. Cashier asks Customer for alternate payment.
3. System receives payment approval and signals approval to Cashier.
   3a. System receives payment denial:
      1. System signals denial to Cashier.
      2. Cashier asks Customer for alternate payment.
4. System records the credit payment, which includes the payment approval.
5. System presents credit payment signature input mechanism.
6. Cashier asks Customer for a credit payment signature. Customer enters signature.

Realistisempi esimerkki: kassapäätejärjestelmä

Tarkkaan kuvattu käyttötapaus

- Esimerkin mallin mukaan käyttötapauksen pääkulku kannattaa kuvata tiiviisti
  - Eri askeleiden sisältöä voi tarvittaessa tarkentaa (askel 7)
- Huomioi tapa, miten poikkeusten ja laajennusten sijainti pääkulussa merkitään
  - 7a $\longrightarrow$ laajentaa/tarkentaa pääkulun kohtaa 7

Osa jossa laajennukset, tarkennukset ja poikkeukset dokumentoidaan, on usein paljon pidempi kuin normaali kulku

- Koska kyse vaatimusmäärittelystä, kuvaus toteutetaan abstraktilla tasolla eli ...
  - Ei oteta kantaa toteutusyksityiskohtiin
  - eikä käyttöliittymään
  - Esim. tunnistetaanko ostos viivakoodin perusteella ...

Yhteenveto käyttötapausmallista

- Käyttötapaukset ovat yksi tapa kuvata ohjelmiston toiminnallisia vaatimuksia
- **Käyttötapauksen tekstuaalinen esitys oleellinen**
- Ohjelmistoprojektissa pitää sopia yhteinen tapa (*käyttötapauspohja*) käyttötapausten tekstuaaliseen esitykseen
- Käyttötapauskaavion merkitys lähinnä yleiskuvan antaja
- Jos huomaat käyttäväsi paljon aikaa "oikeaoppisen" käyttötapauskaavion piirtämiseen, ryhdy välittömästi tekemään jotakin hyödyllisempää (esim. käyttötapausten tekstuaalisia esityksiä)