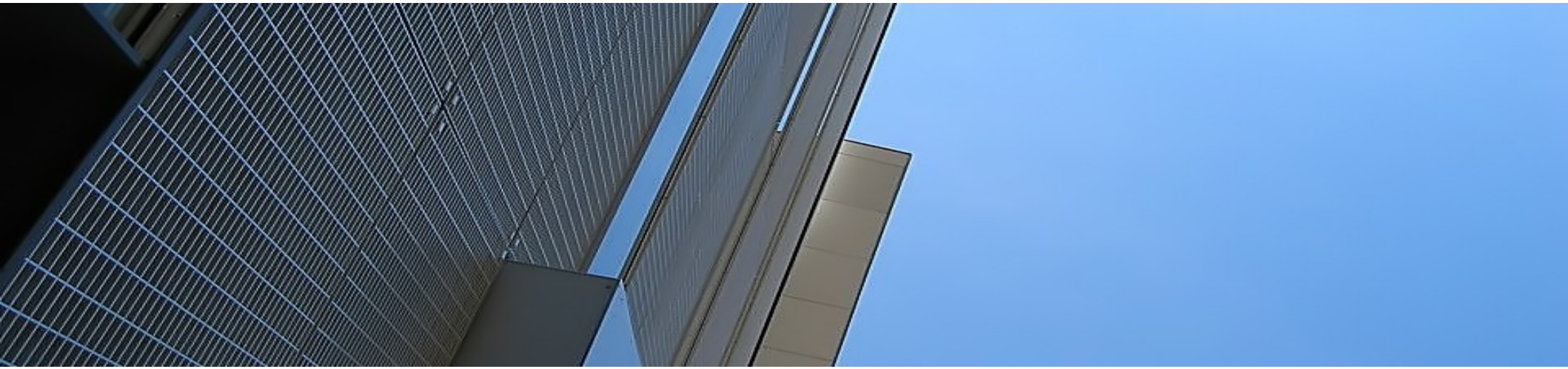
CINCO

Collaborative and interoperable computing



Palvelusuuntautunut ohjelmistotuotanto

Luento 6: Malliperustaisen ohjelmistotuotannon perusteet; palvelutuotannon mallit

Toni Ruokolainen, 5.2.2010

Luennon runko

- Malliperustainen ohjelmistotuotanto
  - Malliperustaisen ohjelmistotuotannon perusteet ja peruskäsitteet
  - Mallit
    - Mitä ne ovat? Miten ne suhtautuvat muuhun ympäristöön? Miten niitä tehdään?
  - Malliperustaisen ohjelmistotuotannon lähestymistapa: OMG:n Model Driven Architecture
    - Metatasot ja abstraktionäkökulmat
    - Tärkeimmät elementit
- Sovellusalueita
- Mallintaminen palvelusuuntautuneessa ohjelmistotuotannossa
  - UML ja UML -profiilit
  - BMM (Business Motivation Model)
  - SoaML (Service oriented architecture Modeling Language)
  - UML4SOA

Ohjelmistotuotannon ongelmakohtia

- Ohjelmistotuotannon monimutkaisuus
  - Ohjelmistojen tulee
    - Toimia heterogeenisissä ja hajautetuissa ympäristöissä
    - Kommunikoida käyttäen erilaisia paradigmoja ja teknologioita (multi-modaalisuus)
    - Mukautua dynaamisesti toimintaympäristön muutoksiin
    - Käyttäytyä luotettavasti (*dependability*)
  - Käsitteellinen kuilu asiakkaan vaatimusten ja ohjelmiston ominaisuuksien välillä
    - Ongelma- ja ratkaisutoimialueen käsitteet ovat eri abstraktiotasolla ja katsovat asioita eri näkökulmista
      - Mitä halutaan? Miksi? vs. Miten?
- Ohjelmistotuotannon tehottomuus
  - Suunnitellut deadlinet vs. Realisoituneet deadlinet
  - Suunnitellut ominaisuudet vs. Toteutuneet ominaisuudet

Malliperustaisen ohjelmistotuotanto (*Model Driven Engineerin,* MDE)

- Kehityshistoria
  - Ohjelmointikielten eri sukupolvet
  - CASE-välineet
  - MDE
  - Trendi: abstraktiotason, tehokkuuden, yleiskäyttöisyyden ja siirrettävyyden lisääminen
- MDE on ”kattotermi” joka sisältää
  - Erilaisia malleja ja niiden käyttötapoja
    - Kehitysmallit (*development models)*
      - Esimerkiksi vaatimus-, arkkitehtuuri-, toteutus- ja asennusmalleja
    - Ajonaikaiset mallit *(runtime models)*
      - Esimerkiksi palvelu-, yhteistoimintaverkosto- ja SLA-mallit
  - Erilaisia lähestymistapoja
    - OMG:n Model Driven Architecture (MDA)
    - Microsofting Software factories
    - Sovellusaluekohtaiset kielet (*domain-specific language*, DSL)
    - Formaalit spesifiointikielet

MDE:n periaatteet

- Malliperustainen ohjelmistotuotanto pyrkii hallitsemaan ohjelmistotuotannon monimutkaisuutta
  - Abstraktiotason nostolla
  - Ohjelmistotuotannon toimintojen automatisoinnilla
- Pääperiaatteet
  - Mallit ovat ohjelmistotuotannon ensisijaisia tuotoksia (vrt. pelkkä dokumentointi)
  - Mallit tuotetaan sovellusaluekohtaista sanastoa käyttäen
    - Käsitteellisen kuilun kaventaminen
  - Järjestelmät kuvataan useita abstraktiotasoja sekä näkökulmia käyttäen
    - Monimutkaisuuden hallinta
  - Ohjelmistot tuotetaan malleista niin pitkälle kuin mahdollista
    - Ohjelmistotuotannon tehostaminen

MDE:n tavoitteet ja hyödyt

- Määrittelyn ja suunnittelun tehostaminen
  - Sovellusalue- ja teknologiaterminologian parempi vastaavuus; sovellusaluekohtaiset kielet (*domain specific language; DSL)*
- Toteutusvaiheen tehostaminen
  - Koodin generointi; toistuvien tehtävien automatisointi
- Suunnitteluartefaktien uudelleenkäytettävyys
  - Käytetään useita abstraktiotasoja sekä näkökulmamalleja järjestelmän mallintamiseen.
  - Teknologiset yksityiskohdat eristetään liiketoimintalogiikan suunnittelusta.
- Toteutusten oikeellisuus
  - Validointi: konformanssi (abstraktiotasojen välillä), konsistenssi (erityisesti näkökulmien välillä)
  - Verifiointi: esim. Mallintarkistus (*model checking*)
- Parhaiden käytänteiden dokumentointi
  - Mallimuunnokset dokumentoivat tuotantometodeja ja -tapoja

Mallit

- Mallit ovat ”todellisuuden” abstraktioita, jotka on muodostettu jotain tarkoitusta varten, jostain näkökulmasta katsottuna
- Malli on jonkin järjestelmän kuvaus
  - Preskriptiivinen malli
    - Esim. suunnittelumallit ja muut spesifikaatiot
    - Määrittelevät kuinka järjestelmä tulee toteuttaa ja minkälaisia ehtoja sen tulee täyttää
  - Deskriptiivinen malli
    - Esim. ontologiat
    - Kuvaavat jo olemassa olevaa järjestelmää
    - Hyödyllisiä analysointiin ja uudelleentoteutuksiin (*re-engineering*)
- Esimerkkejä malleista
  - Kartat, rakennepiirustukset, matemaattiset mallit, simulaatiot, **ohjelmistotuotannon mallit**, ...

Mallin ominaisuuksia

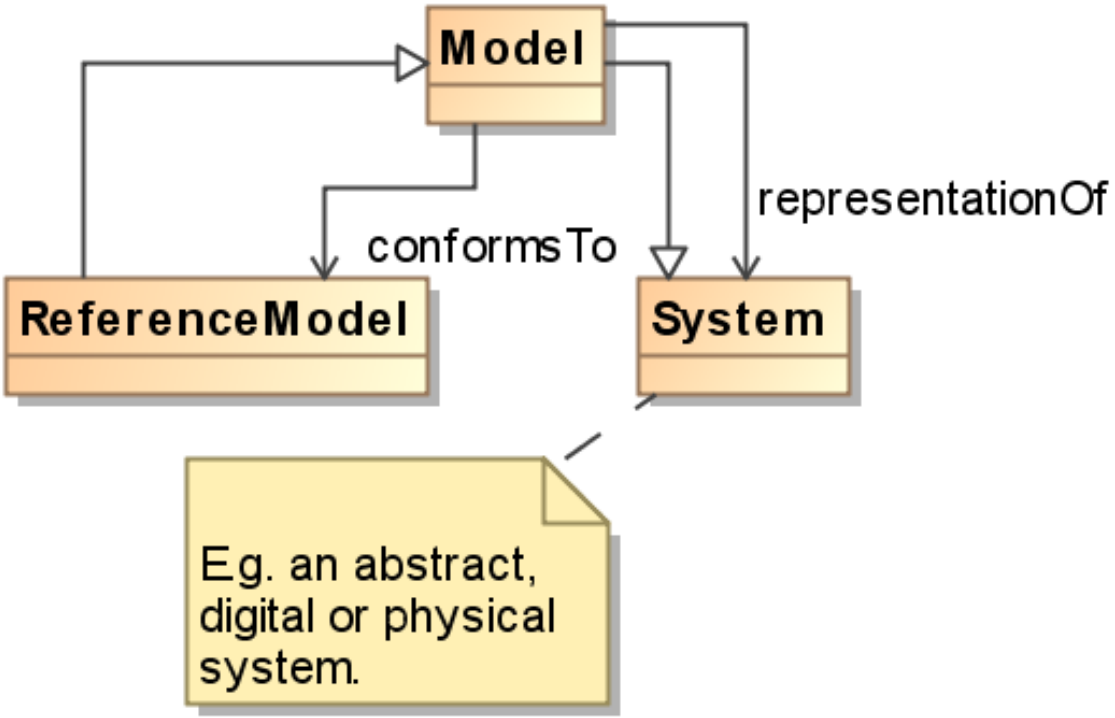
- Antaa kuvauskielen käsitteet
- Määrittelee oikeellisuusehdot kuvauskielen elementeille
- Abstrakti: vain ongelmanratkaisun kannalta relevantit aspektit mukana
- Ymmärrettävä: määritelty sidosryhmien ymmärtämällä tavalla (sanasto, esitystapa, semantiikka)
- Tarkka: kuvaa järjestelmän ominaisuuksia todenmukaisesti
- Ennustava: voidaan käyttää tekemään ennustuksia järjestelmän toiminnasta ja ominaisuuksista
- Edullinen: malli on helpompi, nopeampi ja halvempi valmistaa ja siitä saadaan edullisemmin tuloksia kuin järjestelmästä itsestään

Mallien määrittelemisestä ohjelmistotuotannossa

- Tästä lähtien mallilla tarkoitetaan nimenomaan ohjelmistotuotannossa käytettävää mallia
- Malli on tehty jonkin mallinnuskielen avulla
  - Esim. sovellusaluekohtainen kieli tai UML
- Mallinnuskielellä on abstrakti syntaksi sekä konkreettinen syntaksi.
  - Abstrakti syntaksi määrittelee kielen käsitteet ja rakenteen
  - Konkreettisen syntaksin avulla käyttäjät voivat luoda abstraktia syntaksia vastaavia rakenteita
    - Graafinen tai tekstuaalinen konkreettinen syntaksi
    - Mallinnuskielellä voi olla useita konkreettisia syntakseja
- Järjestelmää kuvaava malli voidaan määritellä useaa abstraktiotasoa ja näkökulmaa käyttäen
  - Toiminnalliset ominaisuudet (esim. mallit arkkitehtuurille, staattisille rakenteille, käyttäytymiselle ja rooleille)
  - Ei-toiminnalliset ominaisuudet (esim. QoS-, luottamus-, liiketoiminta- ja politiikkamallit)

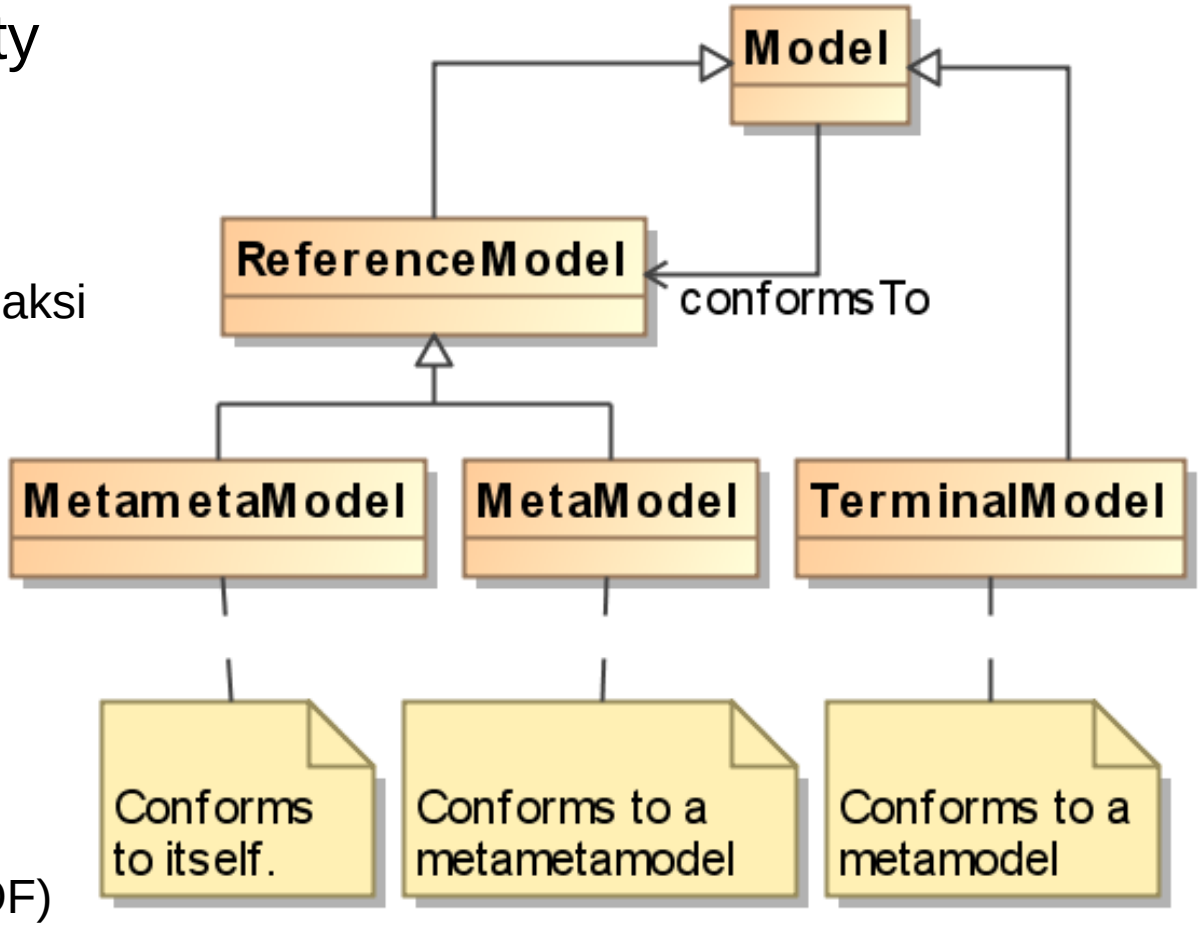
Mallien ja järjestelmien suhteesta

- Järjestelmällä (*System)* tarkoitetaan mitä tahansa mielenkiinnon kohdetta
  - Voi olla esimerkiksi abstrakti (sosiaalinen järjestelmä, ideologia, ontologia), digitaalinen (tietokonejärjestelmä) tai fyysinen (rakennus, silta, kaupunki) järjestelmä
- Malli on kuvaus (*representationOf)* järjestelmästä
  - Malli on itsessään järjestelmä
  - Sisältää joukon toteamuksia (lauseita) järjestelmästä
- Kaikki mallit ovat konformantteja (*conformsTo)* jonkin referenssimallin kanssa
  - Konformanssi tarkoittaa sitä, että mallin jokainen elementti täyttää referenssimallin kyseisentyyppiselle elementille määrittelemät säännöt
  - Referenssimalli määrittelee oikeellisuussäännöt ja abstraktin syntaksin sitä vastaaville malleille



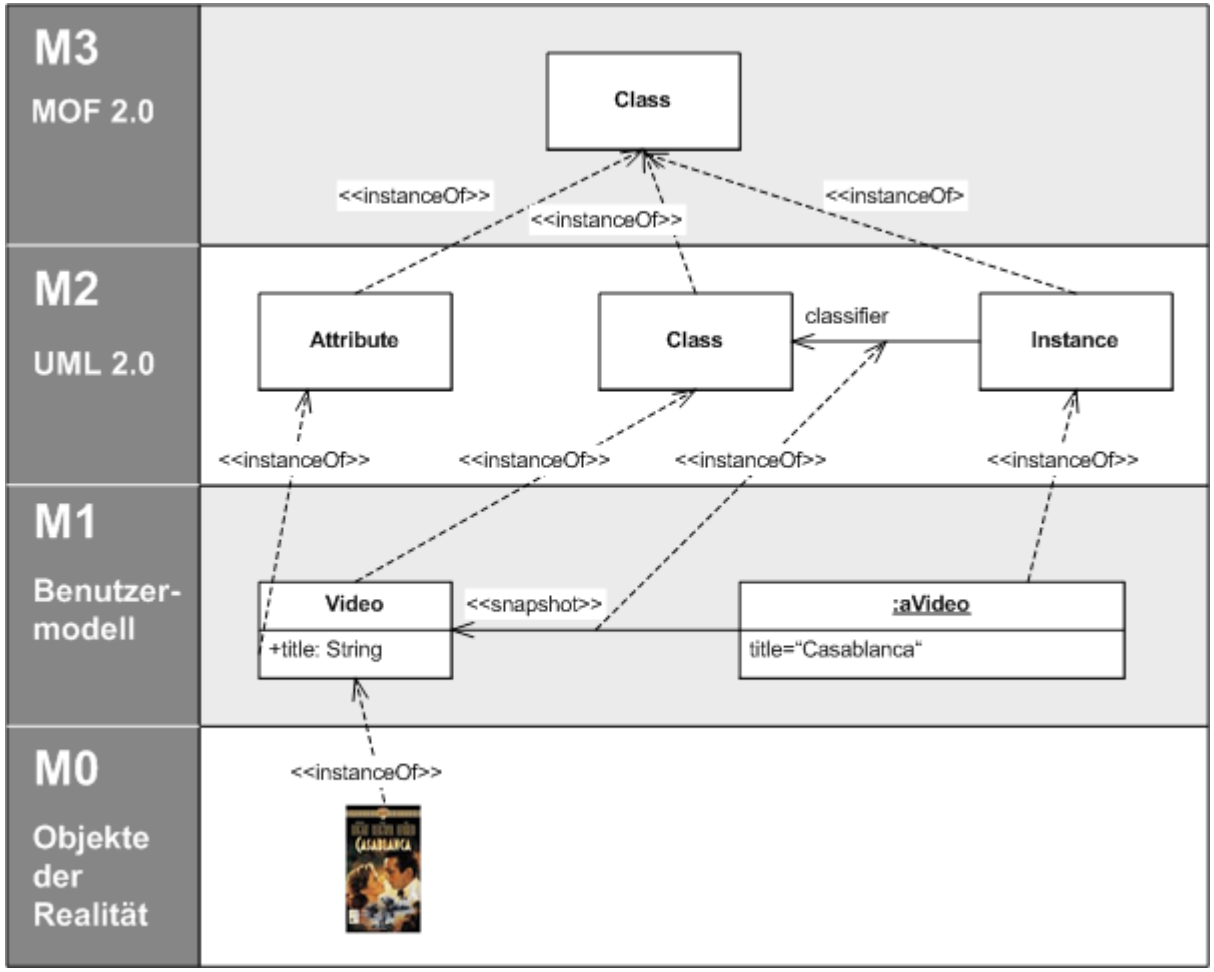
Mallien konformanssi ja kolmikerroksinen metatasohierarkia

- Käytännössä ollaan päädytty kolmikerroksiseen metatasohierarkiaan
  - Kolme kerrosta riittää uusien mallinnuskielien määrittelyn pohjaksi
  - Terminaalimalli on konformantti metamallin kanssa.
  - Metamalli on konformantti metametamallin kanssa
  - Metametamalli on konformantti itsensä kanssa.
- Erilaisia metametamalleja
  - OMG:n Meta Object Facility (MOF)
  - Eclipse Modeling Frameworking Ecore
  - Atlas tutkimusryhmän KM3 (Kernel MetaMetaModel)



MDA ja mallien abstraktiotasot

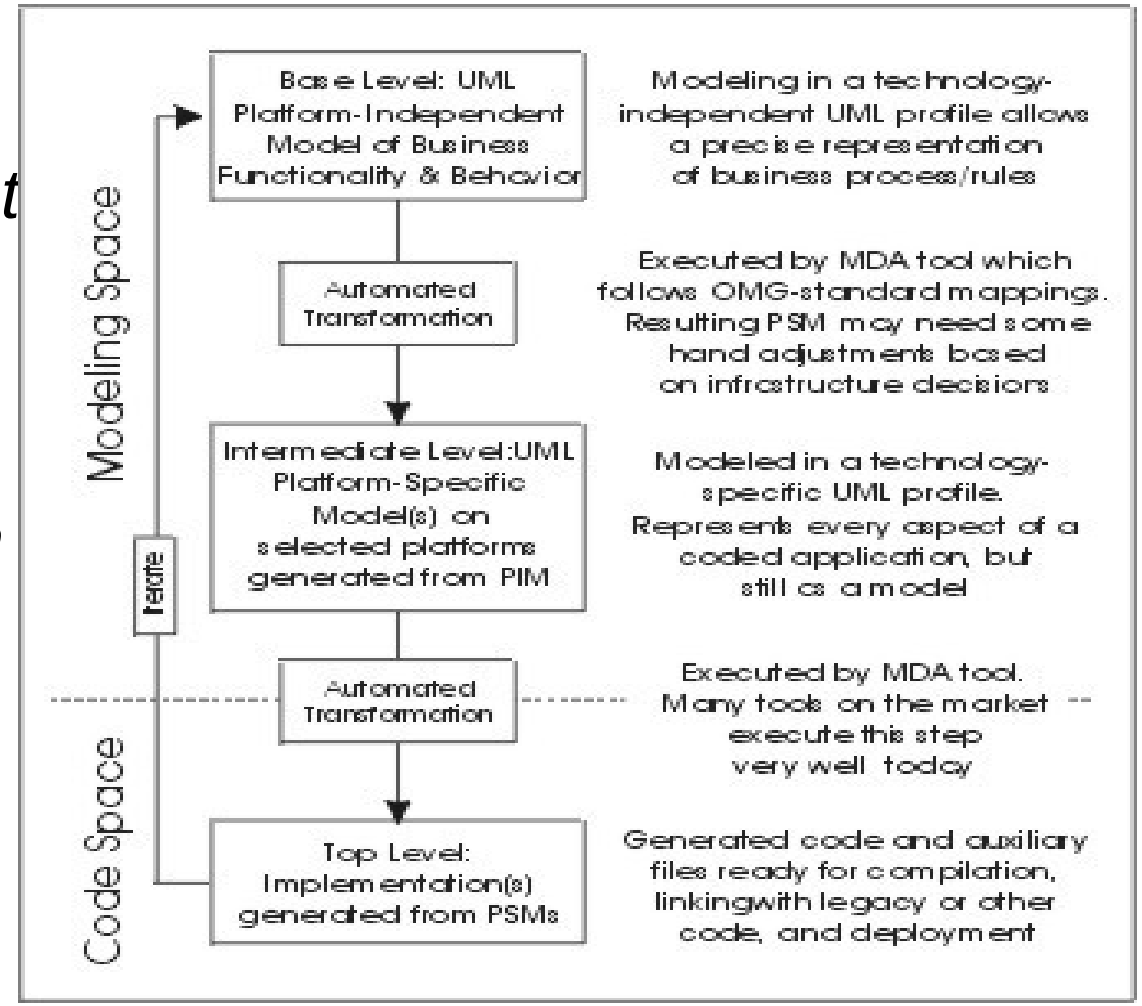
- OMG:n MDA *(model-driven architecture*) standardointikehys on tällä hetkellä yleisin ”metamallinnuskehikko”
- Määrittelee kielen metamallien tekemiseen
  - Sisältää käsitteitä kuten *Class, Property* tai *Operation*
- Neljä abstraktiotasoa:
  - M3: meta-metamalli (MOF)
  - M2: metamallit (UML)
  - M1: mallit (luokkakaavio)
  - M0: käyttäjän objektit
- Kuvassa *instanceOf* vastaa *conformsTo* relaatiota



Kuvan lähde: http://en.wikipedia.org/wiki/Meta-Object_Facility
Huom! Kuvassa vääränlainen tulkinta M0:lle: todellisuus ei ole mallin instanssi vaan mallin pitäisi olla todellisuuden kuvaus.

MDA ja mallinnuksen näkökulmat

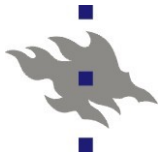
- CIM *(computation independent model)*
- PIM *(platform independent model)*
- PSM (*platform specific model)*
- PDM (*platform description model)*
- Näkökulmien välillä kuljetaan (ylhäältä alas) käyttäen *mallimuunnoksia* ja *mallien kudontaa*



Kuvan lähde: http://www.omg.org/mof/

MDA:n tärkeimmät standardialoitteet

- Metamallien määritteleminen
  - MOF (M*eta-Object Facility)*: metamallin abstraktin syntaksin määritteleminen
  - OCL (Object Constraint Language: metamalliin kuuluvien rajoitteiden määritteleminen
- Mallien kuvaaminen
  - UML (Unified Modeling Language)
  - UML profilointimekanismi: toimialakohtaisten kielien määrittely
- Mallimuunnosten määrittely
  - Mallista-malliin muunnokset: QVT (Query-View-Transformations)
  - Mallista-tekstiin muunnokset: MOFM2T
- Mallien sarjallistaminen ja siirtäminen työkalujen kesken
  - XMI (XML Metadata Interchange)



Esimerkkejä MDE:n sovellusalueista

- Malliperustainen palvelutuotanto
  - Käsitellään kurssin aikana erityisesti harjoitustöiden yhteydessä
- Sovellusten asentaminen (*deployment)*
  - Asennusriippuvuuksien määrittely, hallinta sekä asennuksen automatisoiminen
- Malliperustaiset väliohjelmistot
  - Väliohjelmiston konfigurointi, personalisointi ja spesialisointi tiettyä tarkoitusta varten.
- Malliperustaiset ohjelmistotuotelinjat (*model-driven software product lines)*
  - Ohjelmistotuotelinjojen määritteleminen, toteuttaminen ja hallinta

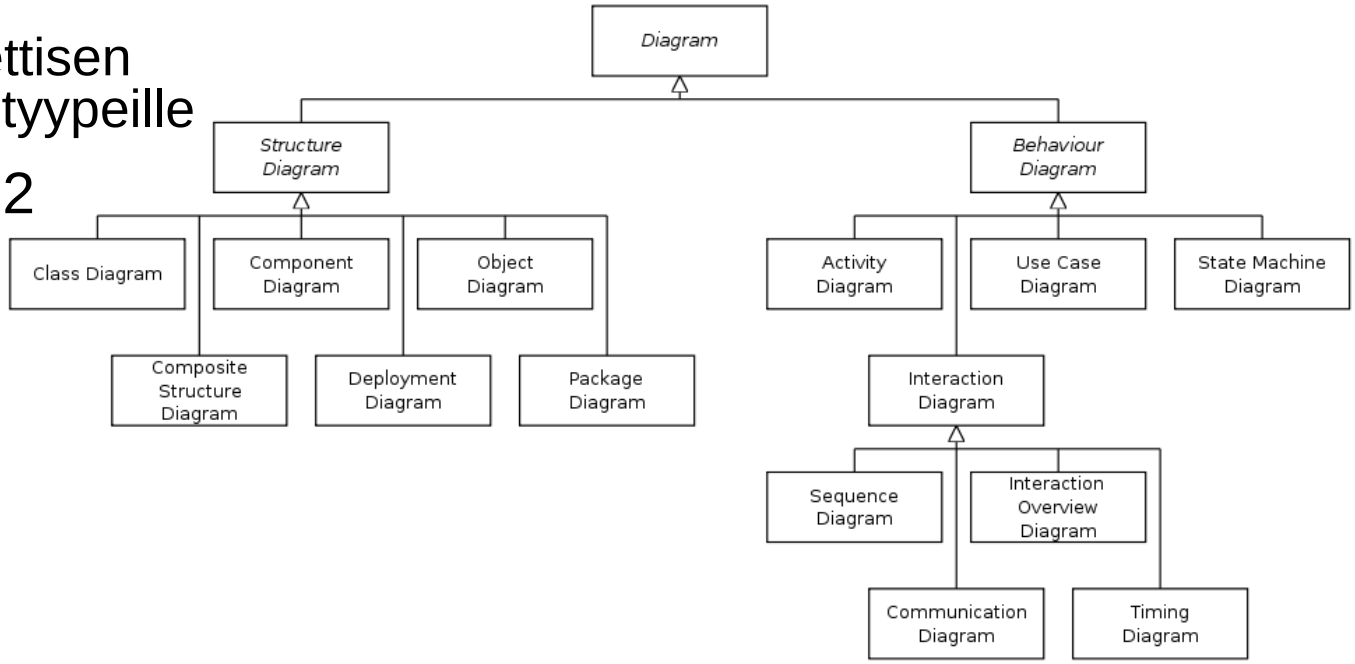
Tauko

Mallintaminen palvelusuuntautuneessa ohjelmistotuotannossa

- Mallien käyttö on erityisen tärkeää palvelusuuntatuneessa ohjelmistotuotannossa
  - Vaatimusten keräämiseen ja määrittelyyn sovellusaluekohtaisilla kielillä
    - Käsitteellisen kuilun ylittäminen
    - Mahdollisesti useiden sidosryhmien tarpeiden kartoittaminen
  - Metainformaation tuottamiseksi ekosysteemin elinkaarien tueksi
  - Yhteentoimivuuden mahdollistamiseksi
    - Jaettujen, yhtenäistävien mallien hyödyntäminen
  - Hajautettujen ohjelmistotuotantoprosessien mahdollistamiseksi
- Mallintamiseen käytetään suuressa määrin UML -pohjaisia työkaluja
  - UML on pitkäikäinen ja tunnettu standardi → kypsät työkalut
  - UML tukee (tiettyyn rajaan asti) sovellusaluekohtaisten kielien tekemistä ja käyttöä
    - UML:n profilointimekanismin avulla UML -työkaluista saadaan DSL -työkaluja

Unified Modeling Language, UML

- UML on OMG:n standardi graafiselle mallinnuskielelle
  - Määrittelee abstraktin syntaksin MOF:in käsitteillä
  - Määrittelee konkreettisen syntaksin eri kaaviotyypeille
- Nykyään versiossa 2.2



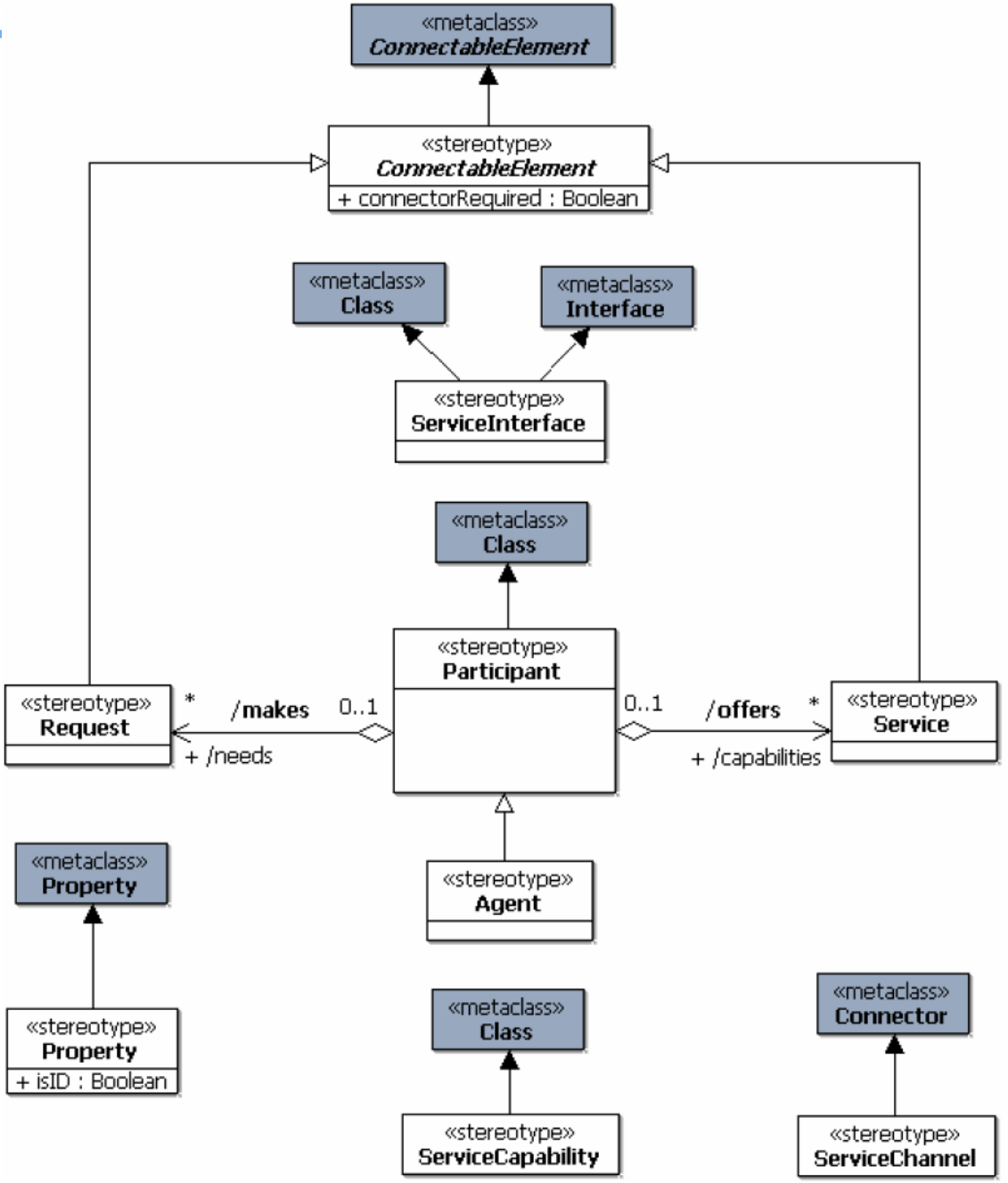
Kuvassa UML 2.0 standardin mukaiset kaaviotyypit.
Kuvan lähde: http://commons.wikimedia.org/wiki/File:Uml_diagram.svg

UML:n profilointimekanismi

- UML:n metamallin luokkia voidaan adaptoida profilointimekanismeilla sopimaan paremmin sovellusalueen tarpeisiin
  - Esimerkiksi tehdään luokkadiagrammeista paremmin J2EE:n käsitteisiin sopivia
- UML profiili pitää sisällään
  - Joukon *stereotyyppejä*
  - Joukon stereotyypeissä määriteltäviä *meta-attribuutteja (tagged value)*
  - Rajoitteita stereotyyppien ja meta-attribuuttien käyttöön liittyen
    - Määritelty OCL -kielellä (*Object Contraint Language*)
  - Mahdollisesti uusia ikoneita stereotyypeille
    - Mahdollistaa sovellusaluekohtaisten notaatioiden käyttämisen
  - Mahdollisesti luonnollisen kielen kuvauksia määriteltyjen käsitteiden semantiikasta

Esimerkki UML -profiilin määrittelystä: SoaML

- SoaML on OMG:n standardialoite palvelukuvauskielelle
- Määrittelee stereotyyppejä (käsitteitä) kuten
  - *Participant*: yhteistoimintaverkostoon osallistuva taho
  - *ServiceCapability*: palvelussa tarjottava toiminto
  - *ServicesArchitecture*: yhteistoimintaverkoston rakenteen kuvaus; määrittelee verkoston osallistujat ja niiden väliset palvelusopimukset



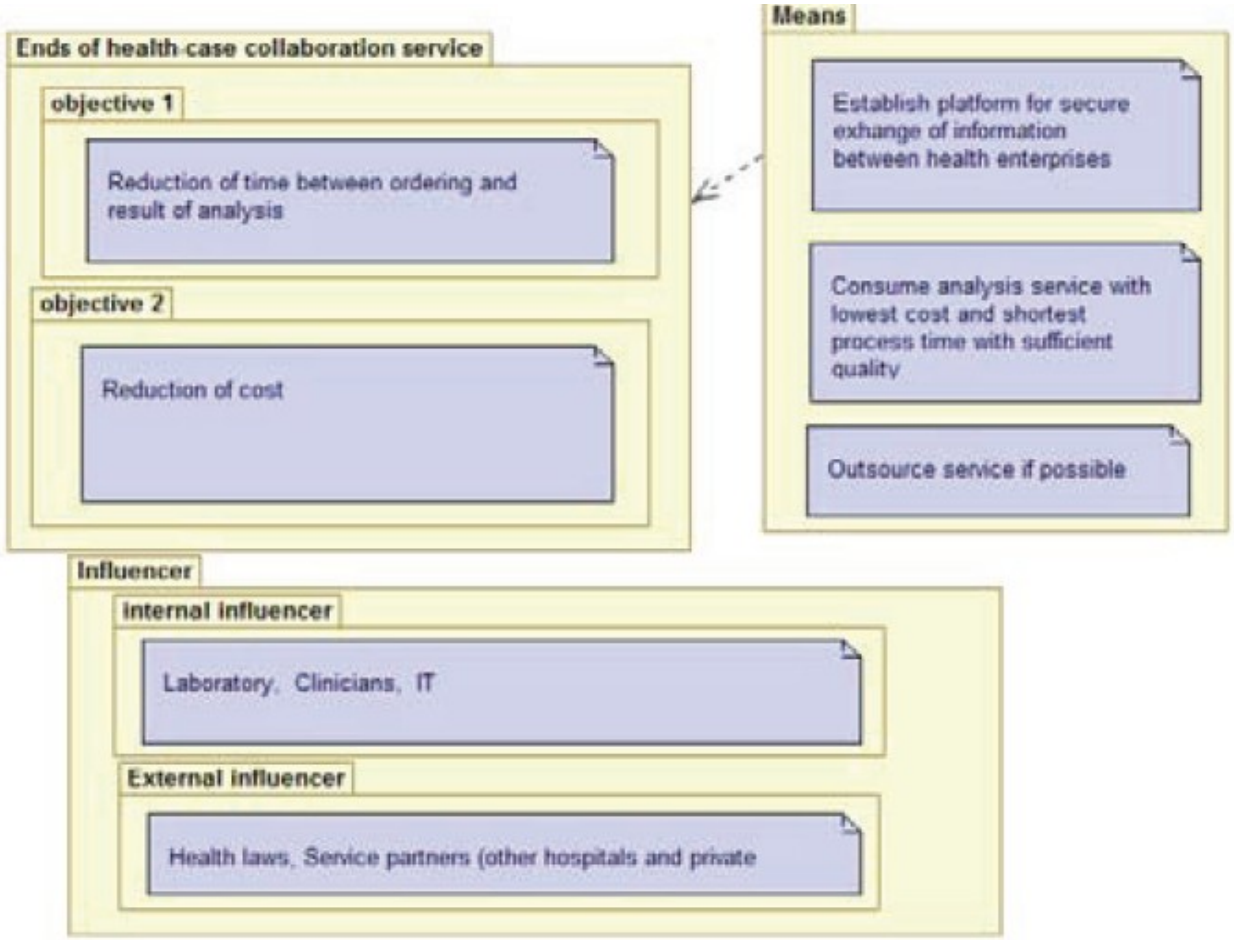
UML -profiileja top-down lähestymistavan mallintamistarpeisiin

- Minkälaisia mallinnuskieliä on käytettävissä top-down palvelutuotannon mahdollistamiseksi?
- Käydään läpi yksi lähestymistapa
  - Perustuu OMG:n standardialoitteisiin sekä niitä vastaaviin UML -profiileihin, SENSORIA -tutkimusprojektin UML4SOA -profiiliin
- Abstraktiotasot ja niitä vastaavat mallit
  - Liiketoimintataso (Business Motivation Model, BMM)
  - Yhteistoimintataso (BPMN, SoaML)
  - Orkestrointitaso (UML4SOA)
- Liikkuminen tasolta toiselle (ylhäältä alas) toteutetaan mallimuunnoksien avulla
  - Mallimuunnoksiin perehdytään luennolla 7

Liiketoimintatason tavoitteiden kuvaaminen: Business Motivation Model

- Business Motivation Model (BMM)
  - OMG:n standardialoite ja siihen liittyvä UML -profiili
  - Käytetään organisaation liiketoimintasuunnitelmien ja -tavoitteiden kuvaamiseen
- Pääkäsitteet
  - *End:* käytetään kuvaamaan organisaation tavoitteita
    - *Goal, Objective,* Vision
  - *Mean:* keinot organisaation tavoitteiden saavuttamiseen
    - *Strategy, Tactic, Business Policy, Business Rule*
  - *Influencer:* vaikuttaja (ulkoinen tai sisäinen) voi vaikuttaa liiketoimintasuunnitelmien tavoitteisiin ja keinoihin
  - *Assesment:* määrittelee vaikuttajan ja liiketoimintasuunnitelman elementtien välisen yhteyden
- Voidaan käyttää yhdessä SoaML:n kanssa
  - Käyttökelpoinen yhdistelmä palveluiden ja niiden tarpeiden identifioimiseen

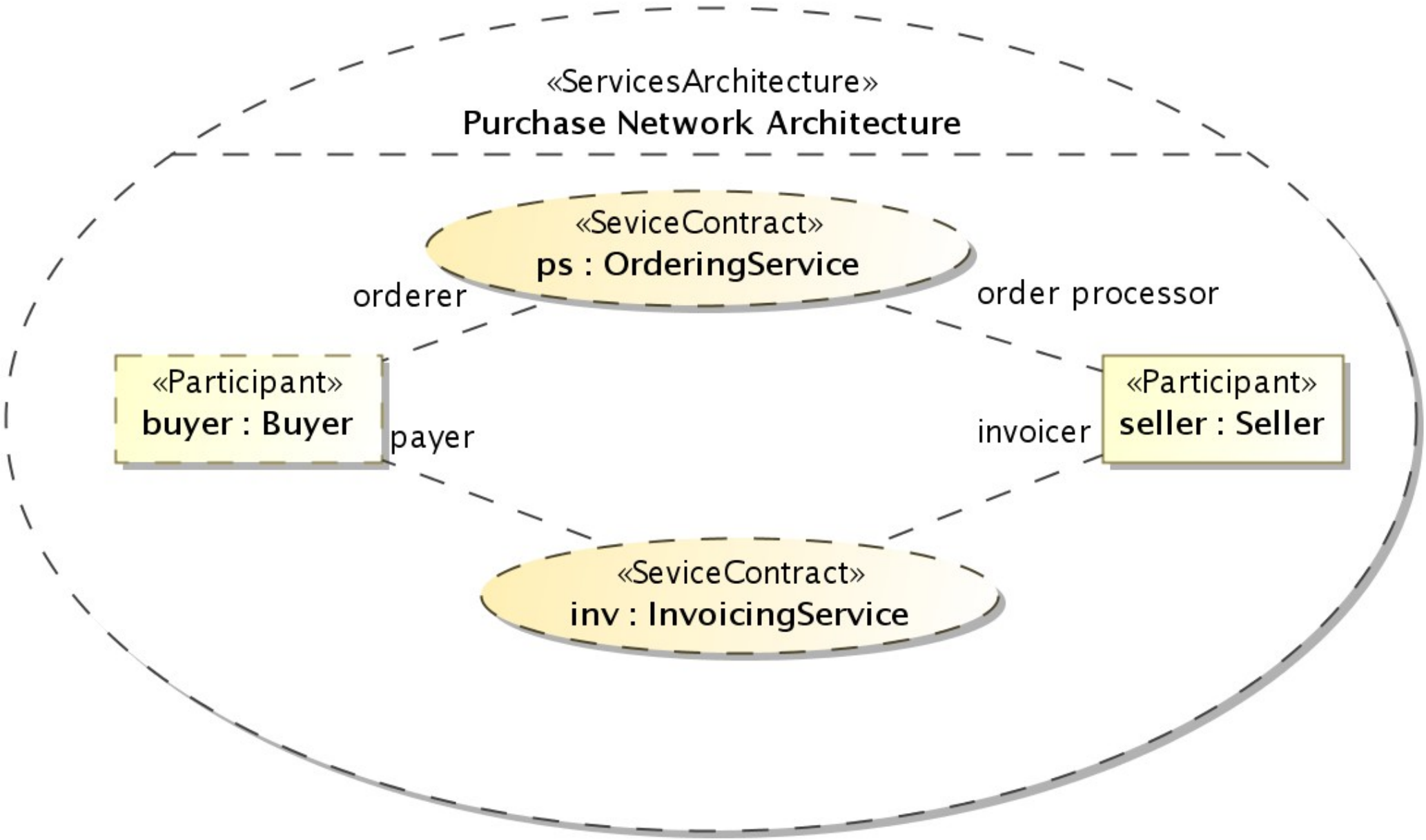
BMM:n käyttöesimerkki



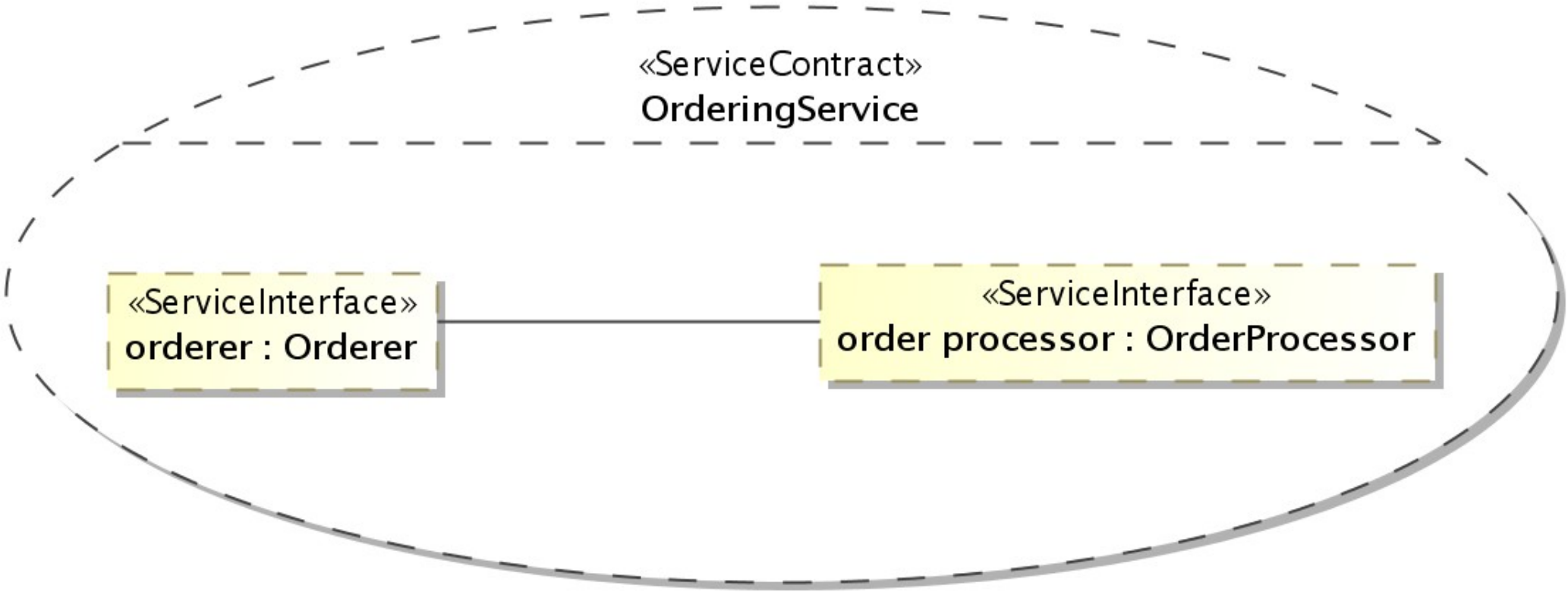
Yhteistoimintatason kuvaaminen: Service oriented architecture Modeling Language

- Service oriented architecture Modeling Language (SoaML)
  - OMG:n standardialoite ja siihen liittyvä UML -profiili
  - Käytetään palveluperustaisten yhteistoimintaverkostojen arkkitehtuurin sekä liiketoimintapalveluiden kuvaamiseen
- Pääkäsitteet
  - *Service:* esittää toimijan kyvykkäättä tuottaa jokin palvelu
  - *ServiceInterface:* kuvaa palvelun rajapinnan; kuvaus sisältää tarjotut ja tarvittavat rajapinnat, toolit, joihin palveluntarjoaja ja asiakas sidotaan ja käyttäytymiskuvauksen (mikä tahansa UML käyttäytymiskuvaus)
  - *Participant:* kuvaa yhteistoimintaverkoston toimijaa
    - Toimija voi olla esimerkiksi ohjelmistokomponentti, organisaatio tai järjestelmä
  - *ServicesArchitecture:* määrittelee palveluperustaisen yhteistoimintaverkoston arkkitehtuurin
    - Kuvaa roolit ja sopimukset, joihin osallistujat sidotaan yhteistoiminnan aikana
  - *ServiceContract:* määrittelee palvelun käytön ehdot, käytettävät rajapinnat ja koreografian

SoaML esimerkki: ServicesArchitecture



SoaML esimerkki: ServiceContract



Orkestrointitason kuvaaminen: UML4SOA

- Orkestrointitasolla kuvataan liiketoimintaroolit täyttävät palveluyhdisteet orkestrointiprosesseina
- Käytetään SENSORIA-projektissa kehitettyä UML4SOA -kieltä
- UML4SOA:lla voidaan kuvata orkestroiniin lisäksi myös
  - Ei-toiminnallisia piirteitä
  - Uudelleenkonfigurointia
  - Politiikkoja
  - Vaatimuksia
- UML4SOA profiilin orkestrointikuvaukset laajentavat UML aktiviteettidiagrammeja

UML4SOA esimerkki

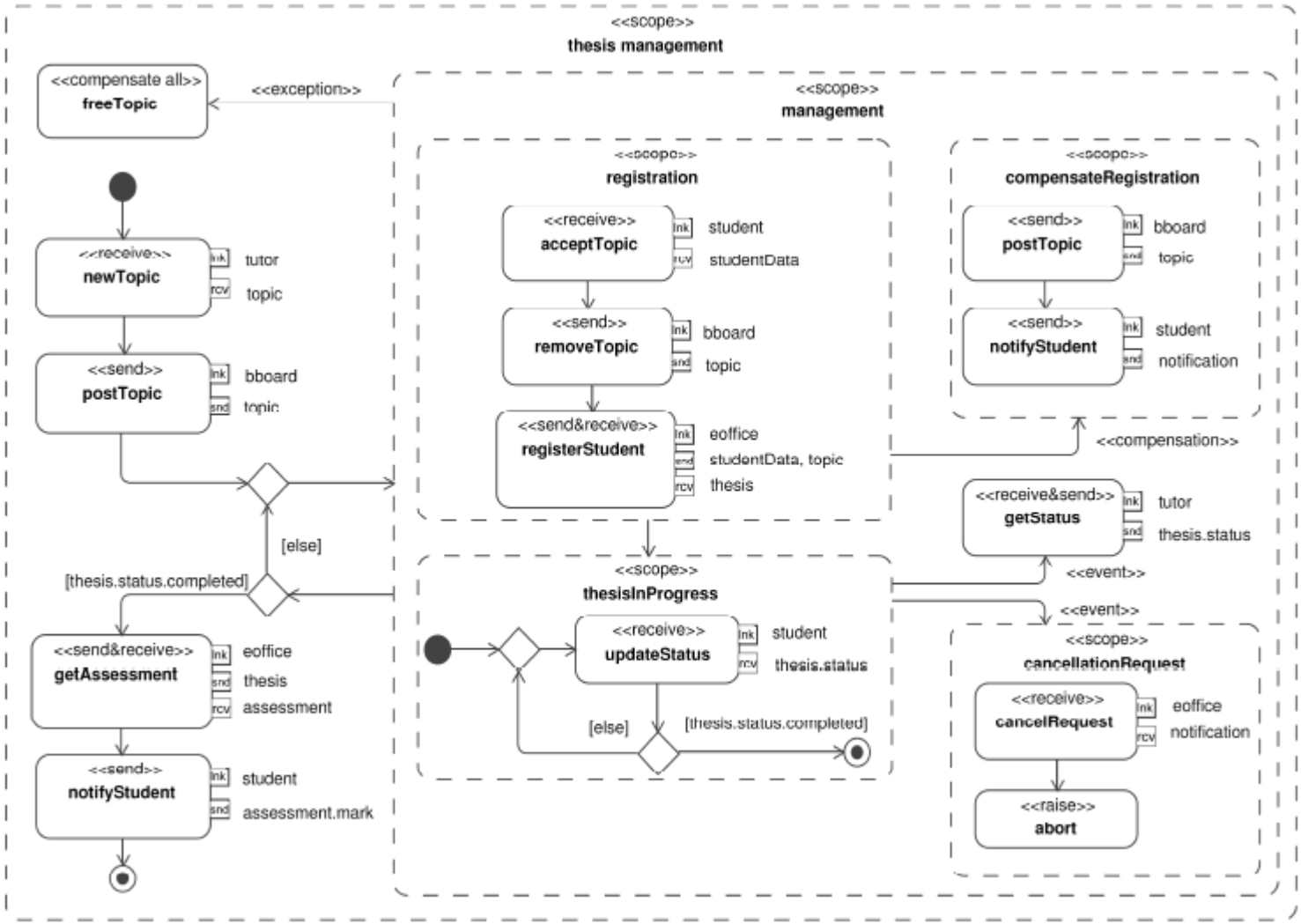


Figure 3: Thesis management modeled with UML4SOA

Entä sitten?

- SoaML- ja UML4SOA profiileiden mukaisesta mallista voidaan tehdä muunnos esimerkiksi
  - Alemman abstraktiotason malleiksi (mallista-malliin muunnoksella)
  - Ohjelmistoartefakteiksi (mallista-tekstiin muunnoksella)
    - SCA (*Service Component Architecture)* teknologian mukaisiin artefakteihin
    - WS-BPEL prosesseihin
    - Java lähdekoodiin
- Tämä tehdään kurssin harjoitustyössä!