



# Profit Group

- loodud 1992. aastal
- töötajaid ~100 (~90% neist tarkvara tegijad)
- kliente ~20 (turuliider Soome kindlustustarkvara turul)
- eesmärgid:
  - ▶ luua konkreetsetele klientidele nende vajadusi rahuldavaid rakendusi
  - ▶ töötada välja täielik kindlustusrakenduste loomist hõlbustav raamistik ja protsess ning vahendid selle kasutamiseks -- Once&Done®
- Once&Done® generatsioonid:
  - ▶ Once&Done® v1 SOM/C++
  - ▶ Once&Done® v2 SOM/C++ & Java (OD EJB server)
  - ▶ Once&Done® v3 Java (teiste tootjate EJB serverid; IBM IAA'le vastavad mudelid)
- osalemine:
  - ▶ IBM IAA litsensi omanik ja osaleb IAA BCB liideste määratlemises (1999)
  - ▶ IBM SF litsensi omanik (1997)
  - ▶ OMG liige



# Once&Done® lühiülevaade

- **Once&Done® ärimudel**
  - ▶ üks lahendus, mis katab kogu kindlustustegevuse äriprotsessi vajadused
  - ▶ üldistatud kindlustustoode
  - ▶ kliendikeskne vaade kindlustuse äriprotsessile
- **Once&Done® süsteemid**
  - ▶ OD raamistikul (kindlustustarkvara sünteesil) põhinevad
  - ▶ Notes'il põhinevad
- **Once&Done® komponendid**
  - ▶ OD mudelid
  - ▶ OD raamistik
  - ▶ OD protsess
  - ▶ OD tööriistad (OD-SE, OD generaatorid)
- **Once&Done® arengusuunad**
  - ▶ kindlustustegevuse areng
  - ▶ tehnoloogia areng



## Once&Done® ärimudel

- üks lahendus, mis katab kogu kindlustustegevuse äriprotsessi vajadused
  - ▶ sama süsteem haldab kõiki kindlustustegevuses vajalike tegevusi
    - info sisestamine, riskide hindamine, lepingu tegemine, lepingu haldamine, kahjude käsitlemine, ...
  - ▶ info sisestatakse vaid üks kord (kui võimalik, siis müügikohas)
  - ▶ info kontrollitakse sisestamisel
- üldistatud kindlustustoode
  - ▶ üldistatud kindlustustoote aluseks on riskimudel, mis katab kõik kindlustuse liigid
  - ▶ kindlustuskatted luuakse vastavalt kliendi vajadusile
  - ▶ riskide hindamine toimub müügiprotsessis
- kliendikeskne vaade kindlustuse äriprotsessile
  - ▶ kliendiga seostatud infot hallatakse ühes kohas
  - ▶ äriprotsessi võib teostada kliendi juures ja tema osalusel



# Once&Done® süsteemid

- OD raamistikul (kindlustustarkvara sünteesil) põhinevad
  - ▶ maksimaalne paindlikus
  - ▶ suured andmemahud
  - ▶ detailne kindlustusmudel
  - ▶ täielik ajalookäsitlus
  - ▶ keerukad (OO) kasutajaliidesed
  - ▶ erinevate tehnoloogiate ühendamine (OO GUI, SQL, EJB, XML, ...)
  - ▶ liidesed välistesse süsteemidesse (MQ, RMI, IIOP)
- Notes'il põhinevad
  - ▶ maksimaalne arenduskiirus
  - ▶ väikesed andmemahud
  - ▶ lihtsustatud kindlustusmudel
  - ▶ lihtsad kasutajaliidesed
  - ▶ Notes'i poolt pakutavad valmisteenused
    - töövoo haldamine, dokumendihaldus, inter-/intra-/extranet'i tugi

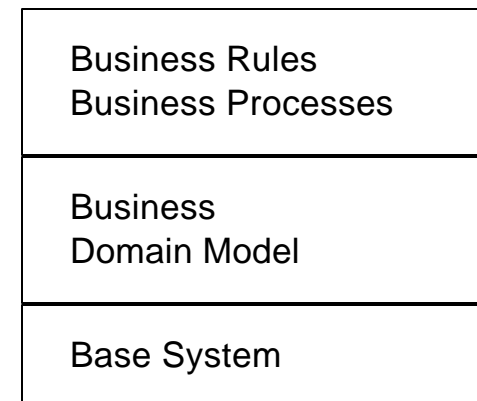


# Once&Done® komponendid

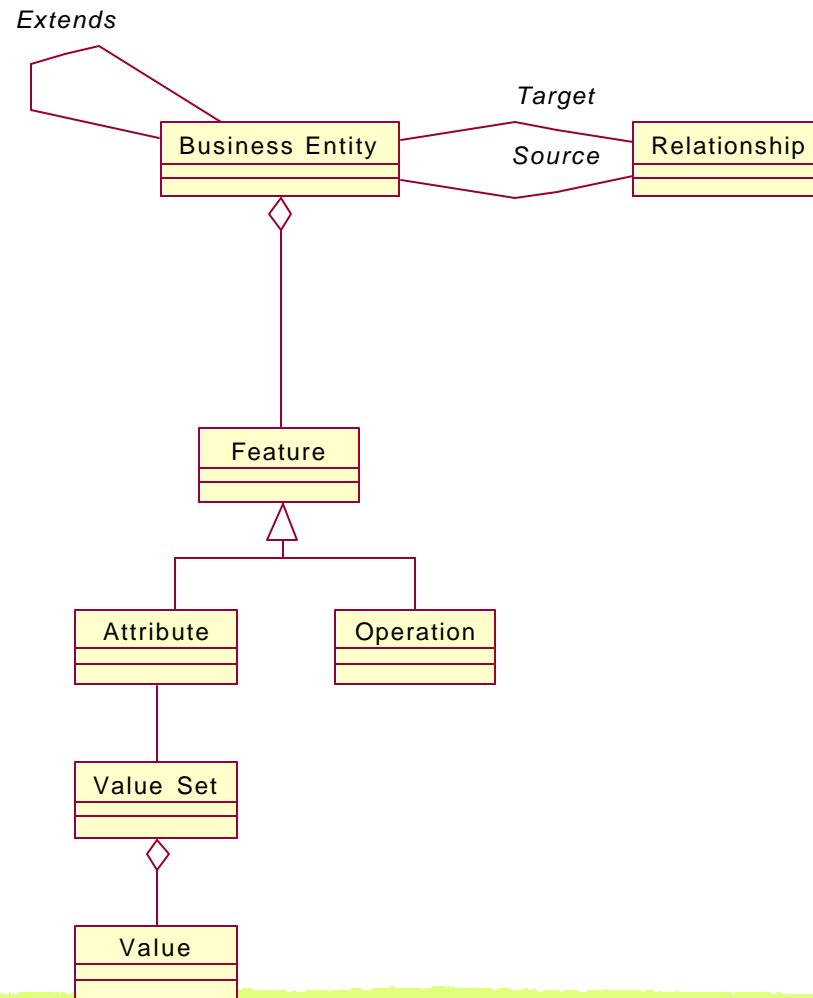
- OD mudelid
  - ▶ kindlustusvaldkonna OO analüüsi meta-mudel (OD-SE meta-mudel)
  - ▶ kindlustusvaldkonna OO analüüsimudelid, mis on Once&Done® süsteemide tegemisel modelleerimise aluseks
- OD raamistik
  - ▶ tehnilised (baas)teenused OO süsteemide ehitamiseks
  - ▶ liidesed Once&Done® süsteemide ärifunktsionaalsuse esitamiseks
  - ▶ OD kindlustusvaldkonna mudelite ja kindlustusfunktsionaalsuse üldistatud realisatsioon
- OD protsess
  - ▶ Once&Done® süsteemide tegemiseks vajalike sammude kirjeldus
- OD tööriistad
  - ▶ vahendid OD protsessi rakendamiseks ja OD raamistiku (ning OD mudelite) kasutamiseks Once&Done® süsteemide tegemisel

# OD mudelid

- Kindlustusvaldkonna OOA/OODmeta-mudel (OD-SE meta-mudel)
  - ▶ üldine OOA/OOD meta-mudel
  - ▶ kindlustusvaldkonnale orienteeritud laiendused OOA/OOD meta-mudelis
  - ▶ tarkvara-arhitektuurile orienteeritud laiendused OOA/OOD meta-mudelis
    - orienteeritud kihilisele modelleerivale äritarkvara arhitektuurile
- Kindlustusvaldkonna OO analüüsimumelid, mis on Once&Done® süsteemide tegemisel modelleerimise aluseks
  - ▶ üldine ärisüsteemide mudel
  - ▶ kliendihalduse mudeli
  - ▶ kahjukindlustuse mudel
  - ▶ elukindlustuse mudel
  - ▶ ...

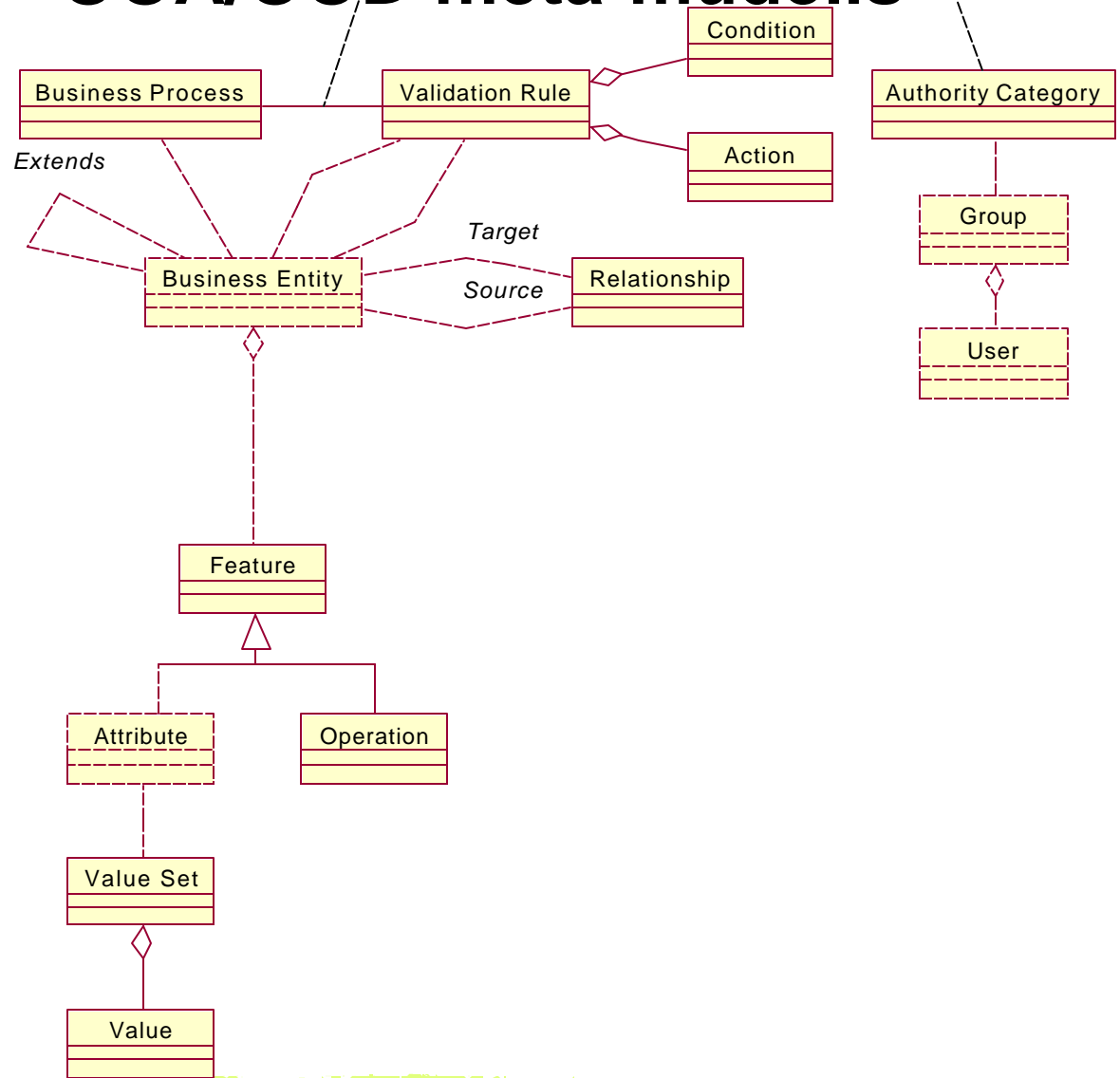


# Üldine OOA/OOD meta-mudel





# Äritarkvarale orienteeritud laiendused OOA/OOD meta-mudel



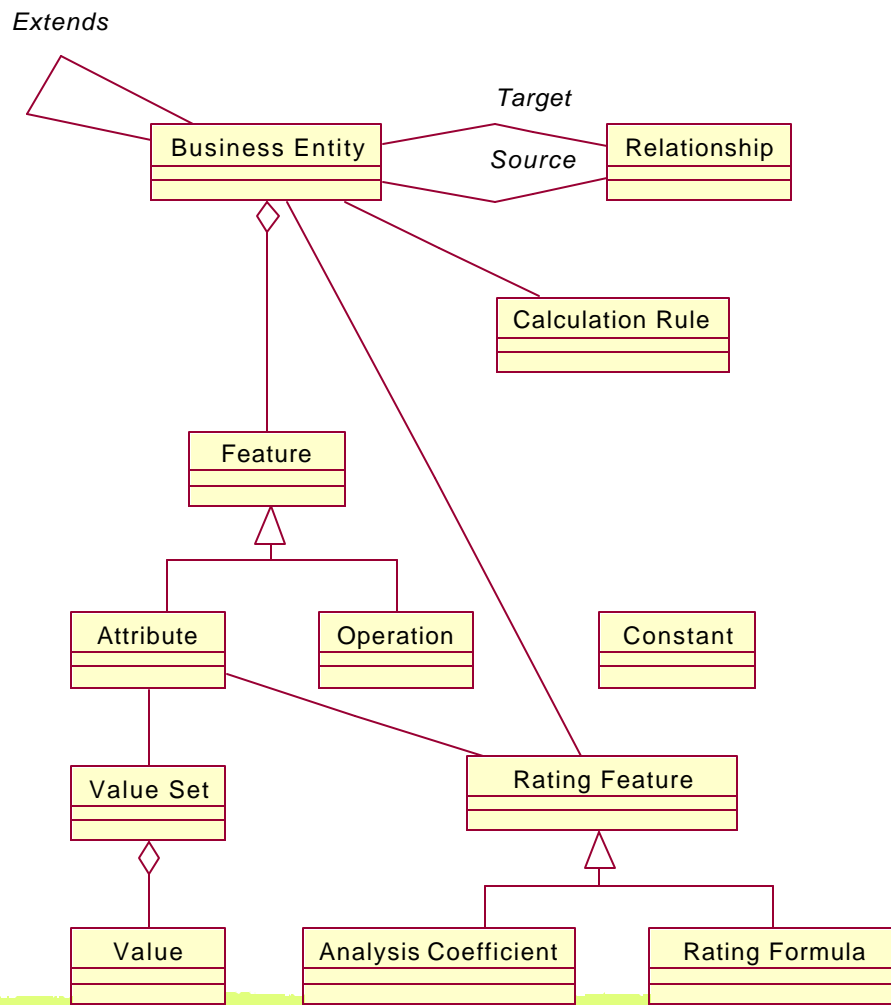




# Äriprotsessid

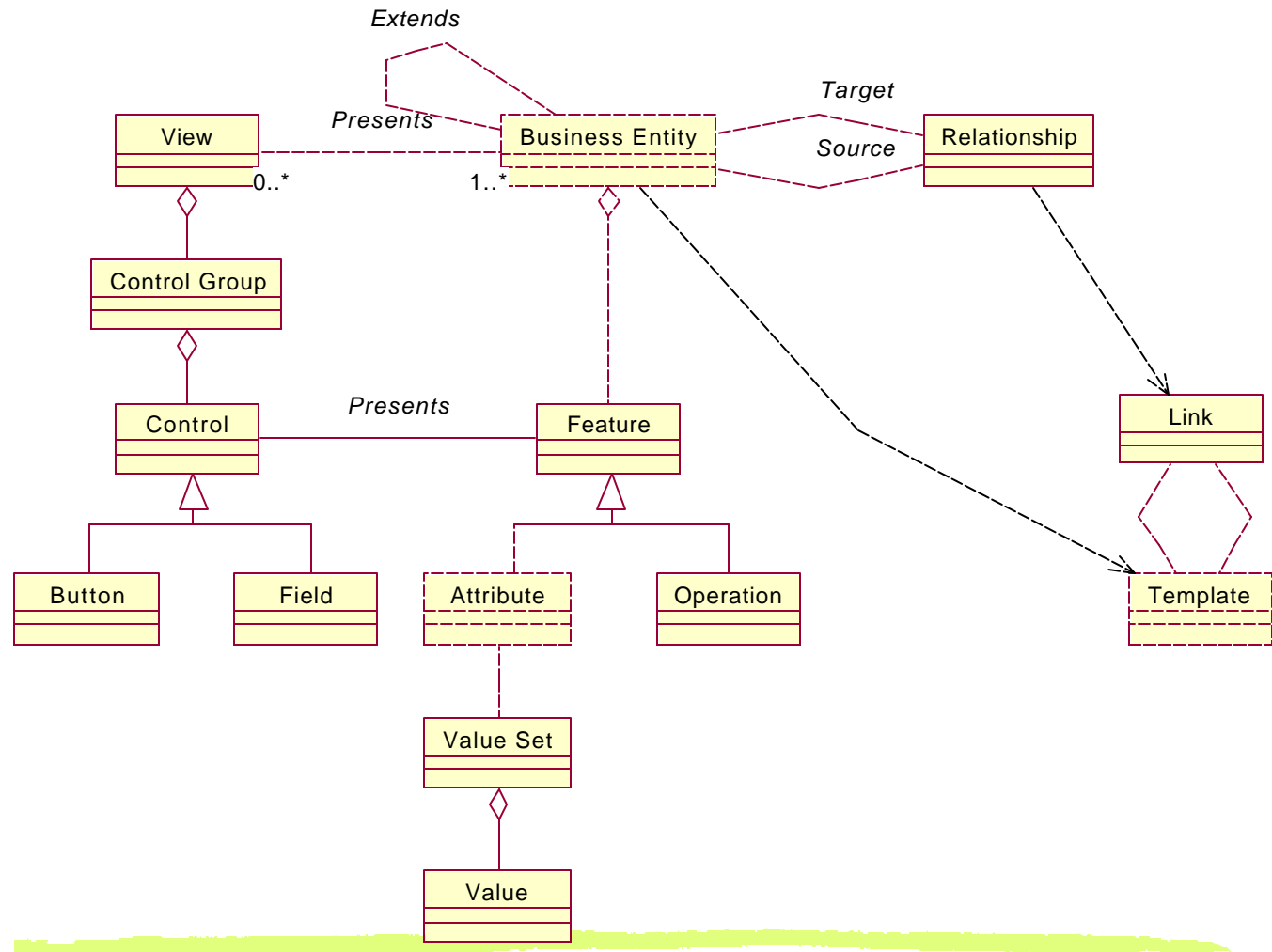
- Äriprotsessi komponendid
  - ▶ ärireeglite hulk
  - ▶ ärivaldkonna objektide hulk
  - ▶ äriprotsessi kirjeldus
- Ärireeglid
  - ▶ ärivaldkonna objekti kohta kehtivad
  - ▶ seostatud ärivaldkonna objektide hulga kohta kehtivad
  - ▶ deklaratiivsed
- Äriprotsessi faasid
  - ▶ valideerimisfaas
    - valdkonna objektide kogumine
    - ärireeglite käivitamine
  - ▶ täitmisfaas
    - äriprotsessi kirjelduse täitmine

# Kindlustusvaldkonnale orienteeritud laiendused OOA/OOD meta-mudelis

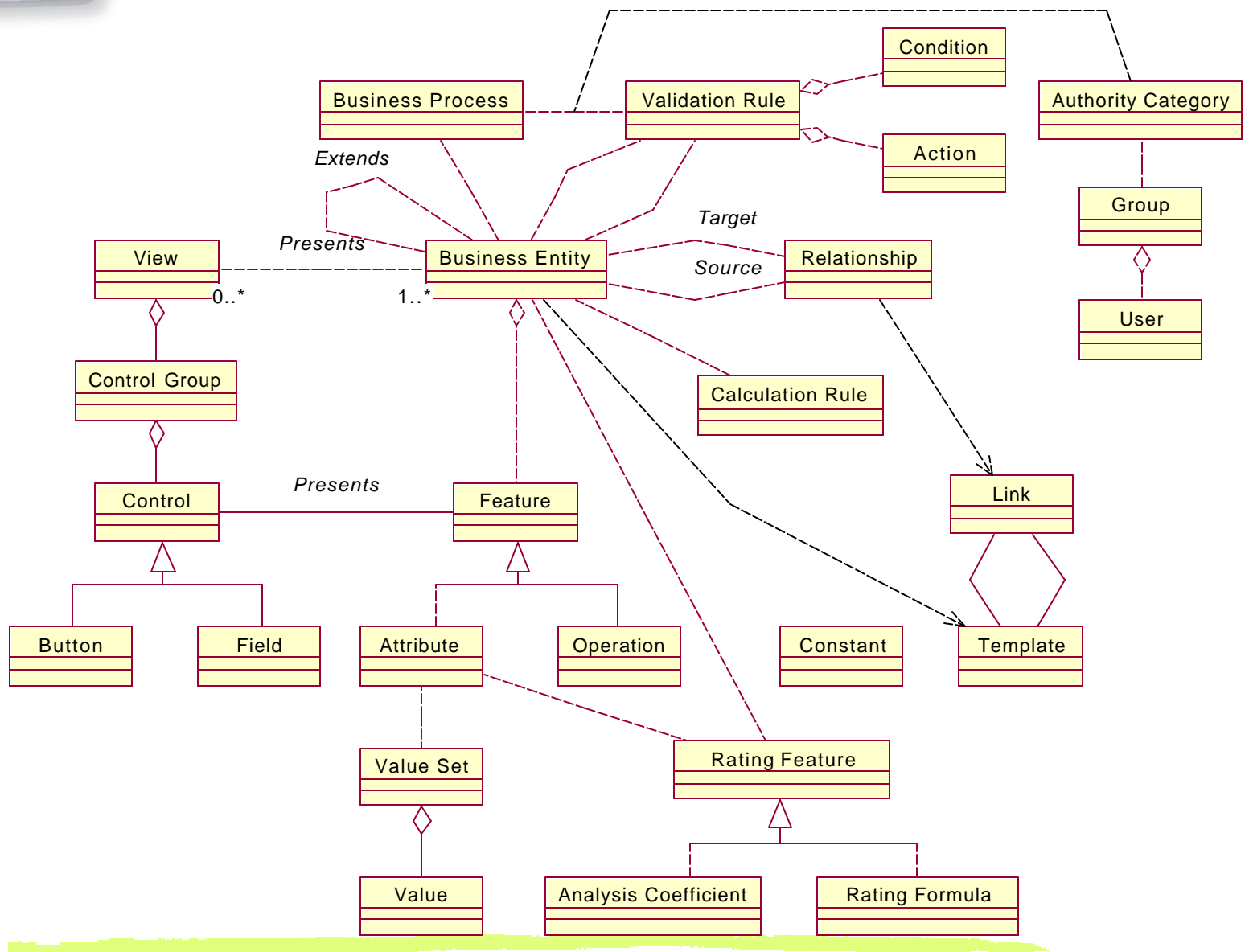




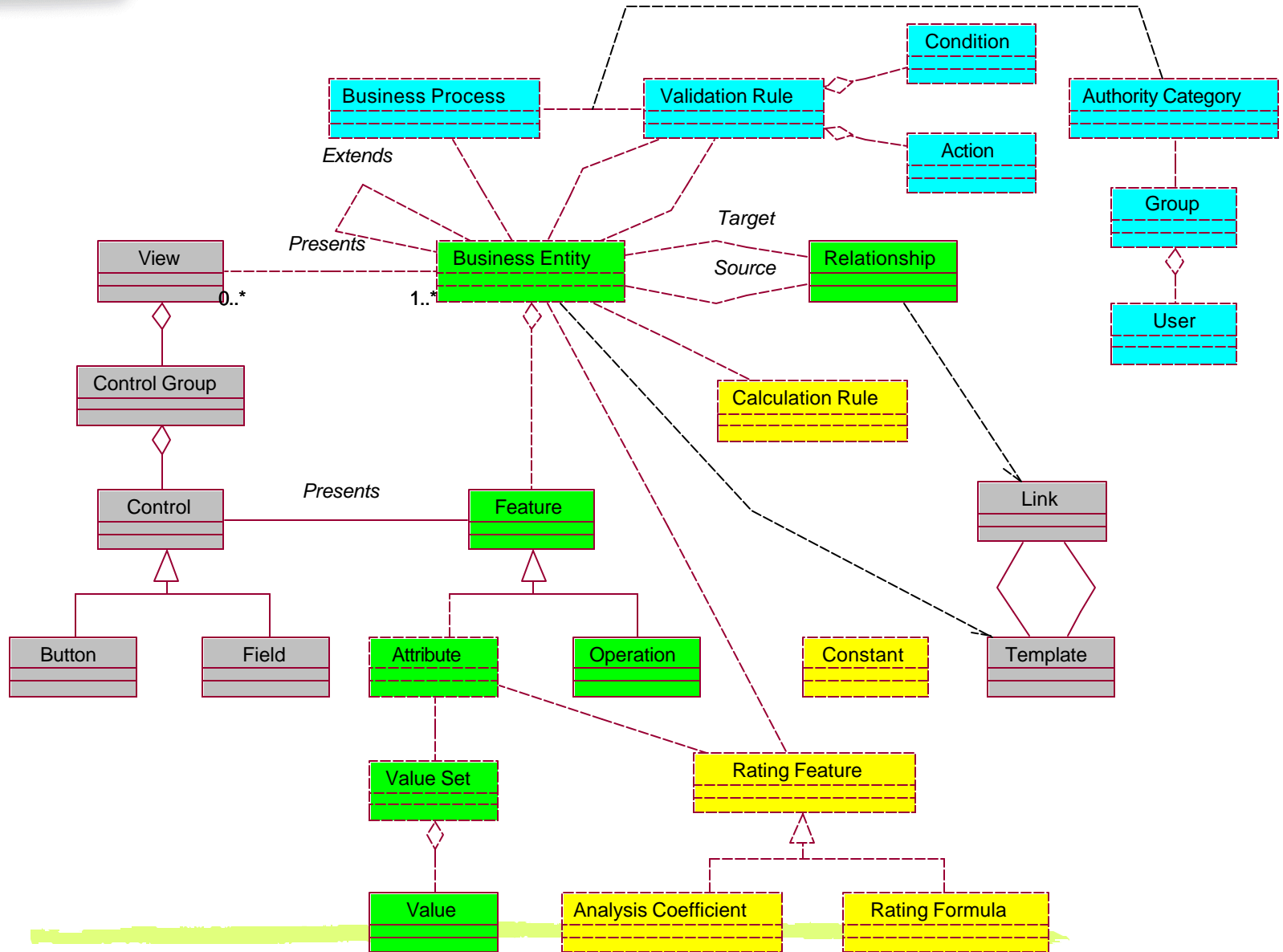
# Tarkvara-arhitektuurile orienteeritud laiendused OOA/OOD meta-mudelile



# Laiendatud OOA/OOD meta-mudel

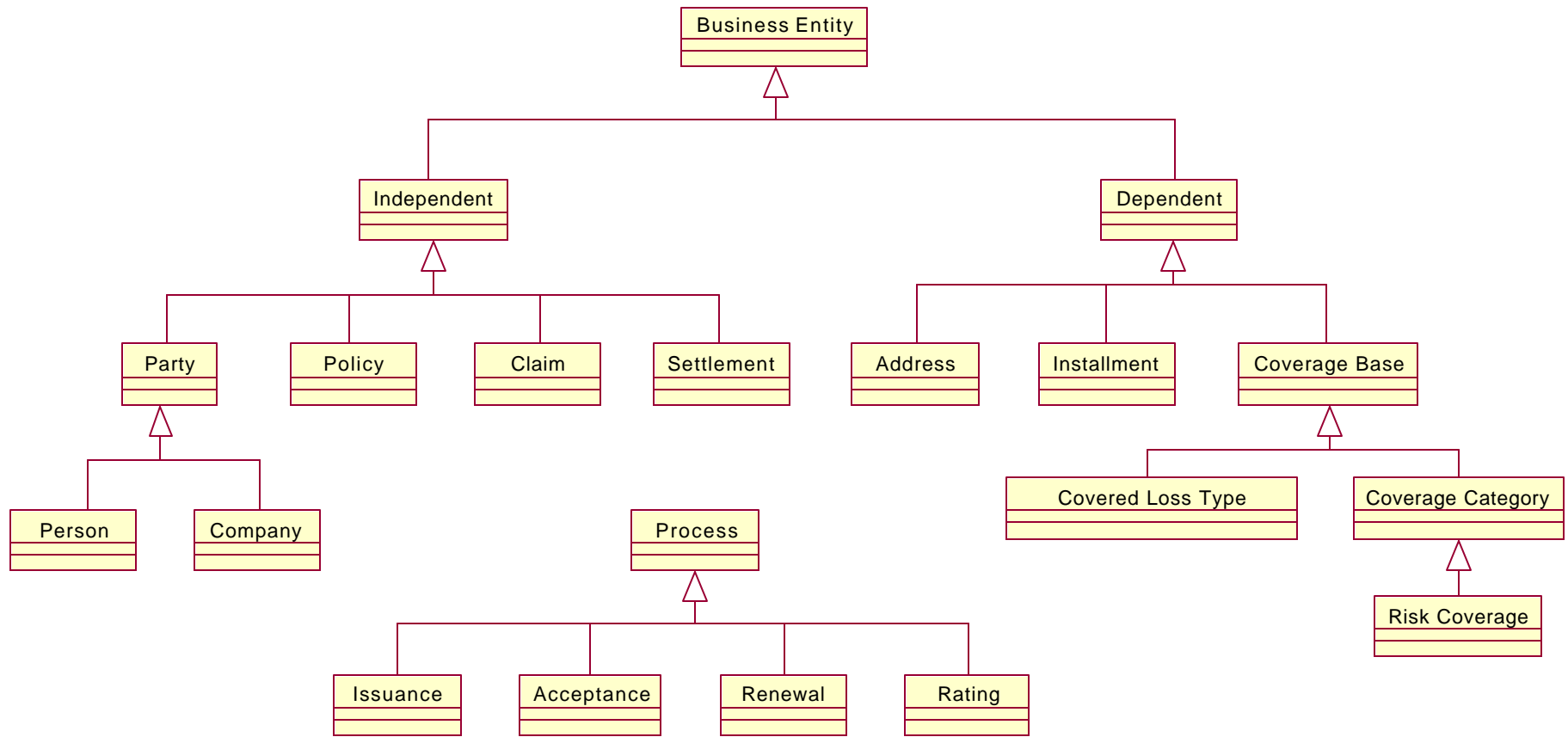


# Laiendatud OOA/OOD meta-mudel





# Kindlustusvaldkonna viitemudel





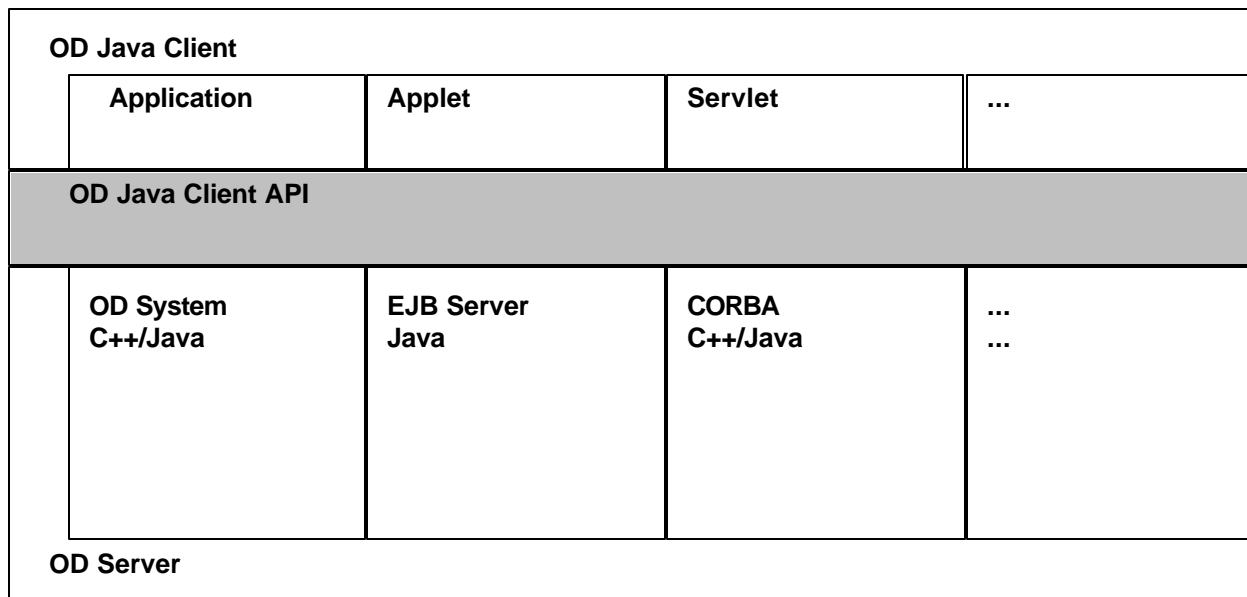
# OD raamistik

- Tehnilised (baas)teenused OO süsteemide ehitamiseks
  - ▶ nime-, sündmus-, püsivus-, suhte-, päringu-, tehingu-, sisevaatlus- (meta-info-), elutsükli-, arvutus-, andmete pärivus- (levitus-), reegli-, objektide vahetus-, esitlus-, väljatrüki-, side- ja otsinguteenus
- Liidesed Once&Done® süsteemide ärifunktsionaalsuse esitamiseks
  - ▶ OD Client API
  - ▶ OD Server API (Java jaoks on valitud EJB standard)
- OD kindlustusvaldkonna mudelite ja kindlustusfunktsionaalsuse üldistatud realisatsioon



# OD Client API

- sama API ning sama programmeerimismudel kliendi ning serveri koodi kirjutamiseks
- tehnoloogiast sõltumatus
  - ▶ võib olla realiseeritud OD objektidel, EJB'del, CORBA objektidel või lokaalsetel objektidel





# OD Client API põhielemendid

## ■ ODSession

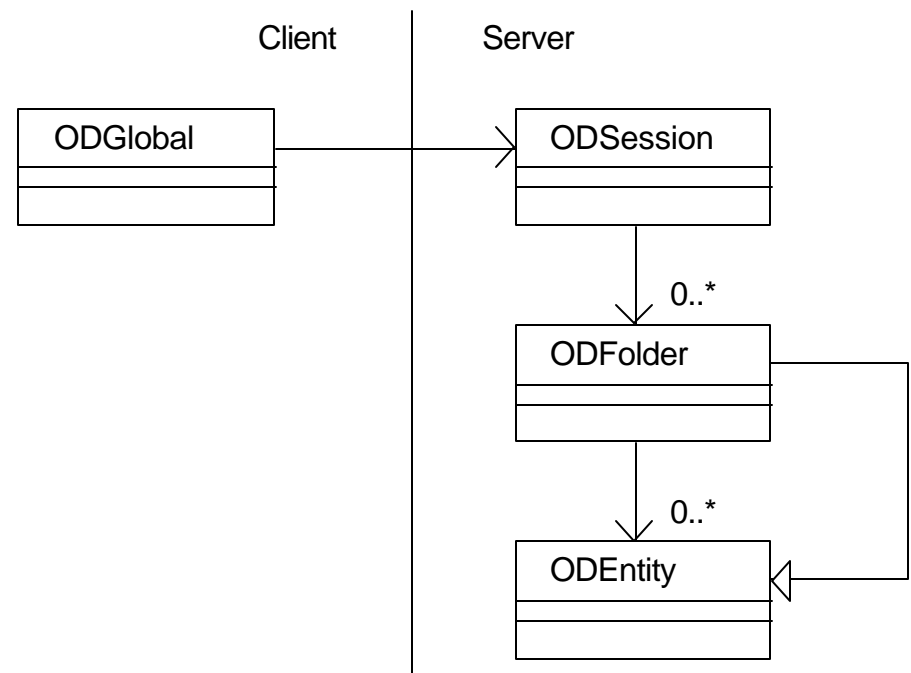
- ▶ kliendi sessioon
- ▶ kaustade "kodu" (haldab kaustade elutsükli)
- ▶ haldab äritehinguid

## ■ ODEntity

- ▶ iseseisev püsiv ärialdkonna objekt
- ▶ haldab navigeerimist (teiste temaga seotud objektide leidmist)

## ■ ODFolder : ODEntity

- ▶ iseseisvate püsivate (üksteisega seotud) ärialdkonna objektide hulk
- ▶ iseseisvate püsivate ärialdkonna objektide "kodu" (haldab nende elutsükli)



# OD süsteemi konfiguration

## ■ Mudel

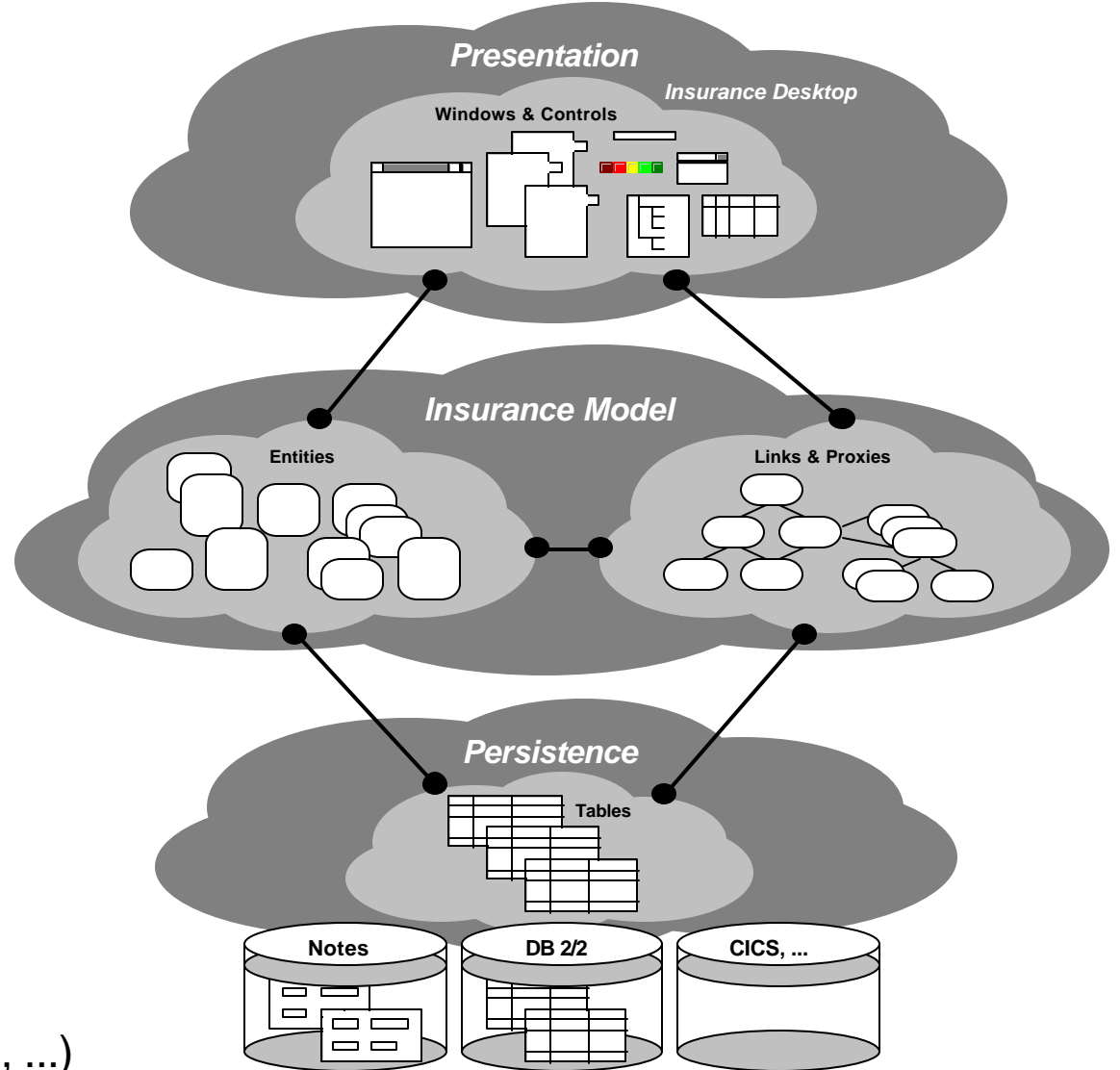
- ▶ olemid ja asemikud (*proxy*)
- ▶ seosed (suhted)

## ■ Esitus

- ▶ vaated (aknad)
- ▶ vaateelemendid (väljad, ...)
- ▶ dokumendid (väljatrükid)

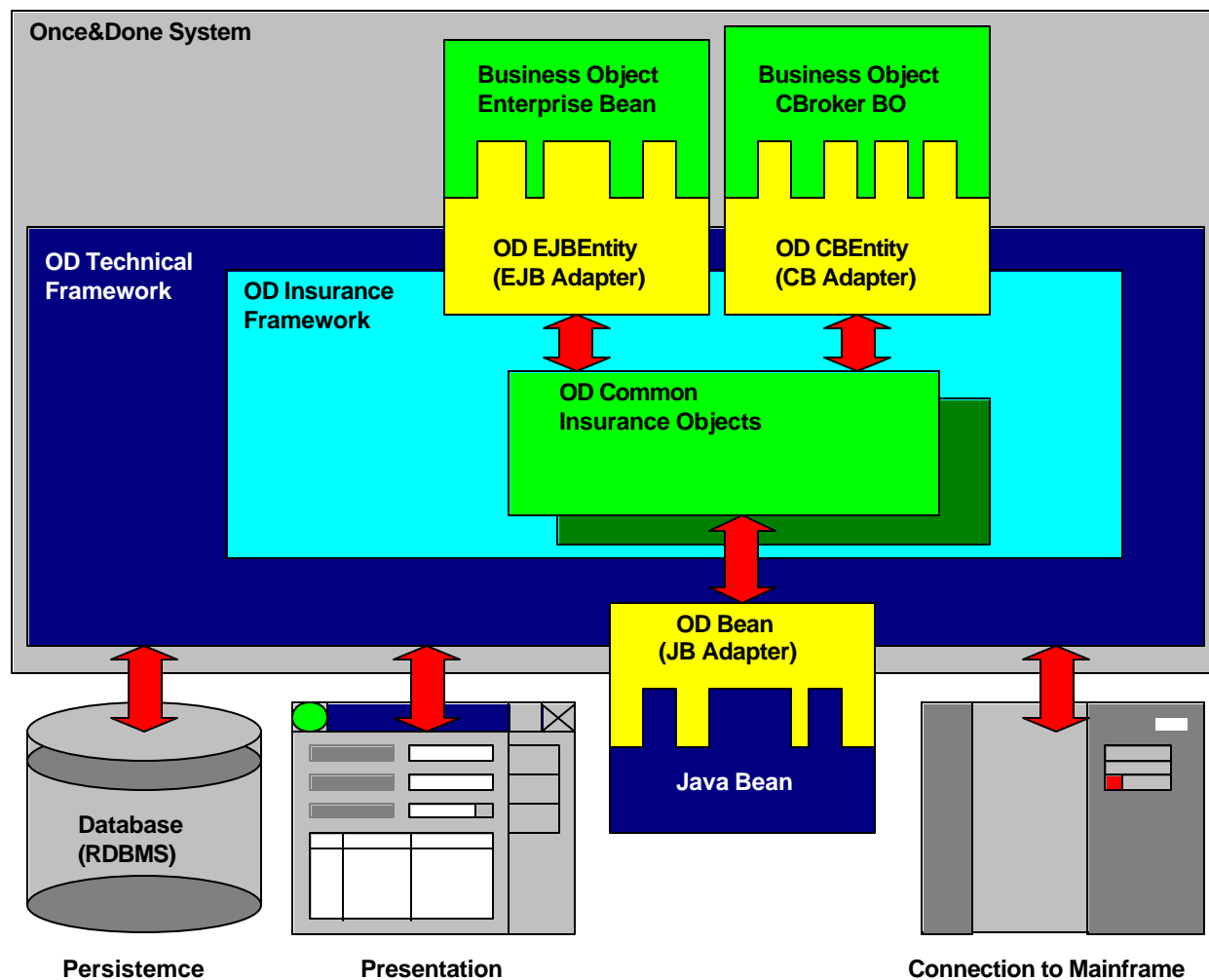
## ■ Teenused

- ▶ tehinguteenus
- ▶ püsivusteenus
- ▶ sideteenus
- ▶ reegliteenus
- ▶ sündmusteenus
- ▶ suhte- ja päringuteenus
- ▶ kindlustusteenused (hindamis-, ...)
- ▶ ...



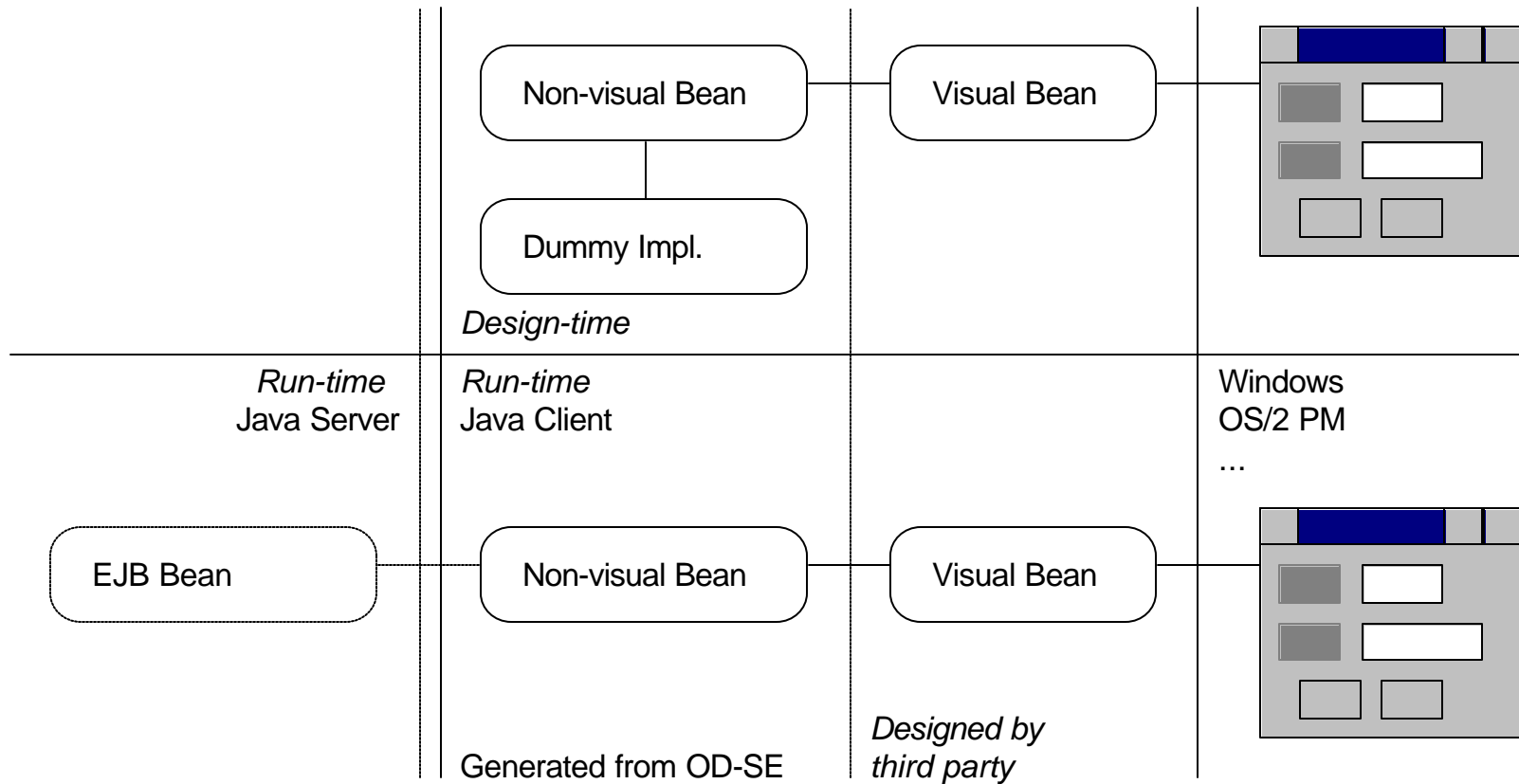


# OD süsteemi tehniline arhitektuur



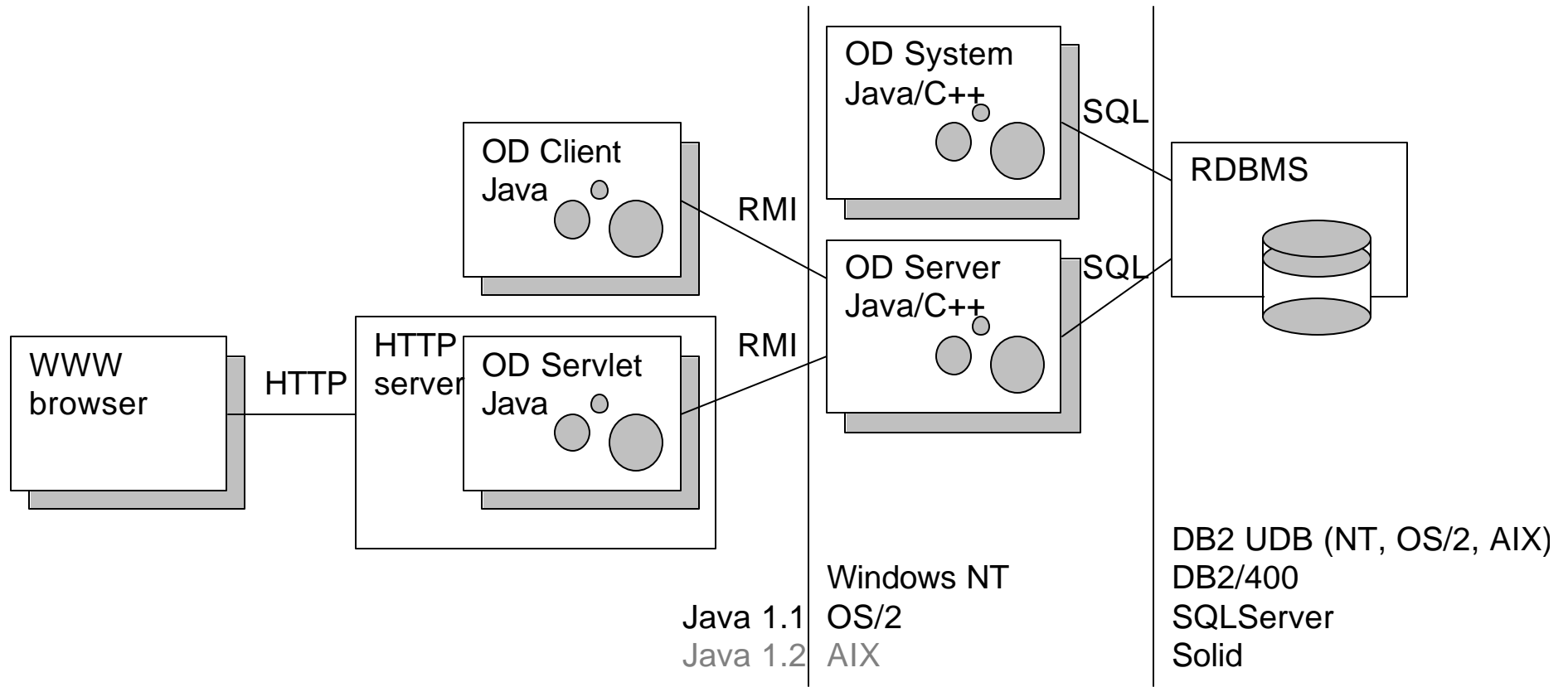


# OD GUI Raamistik



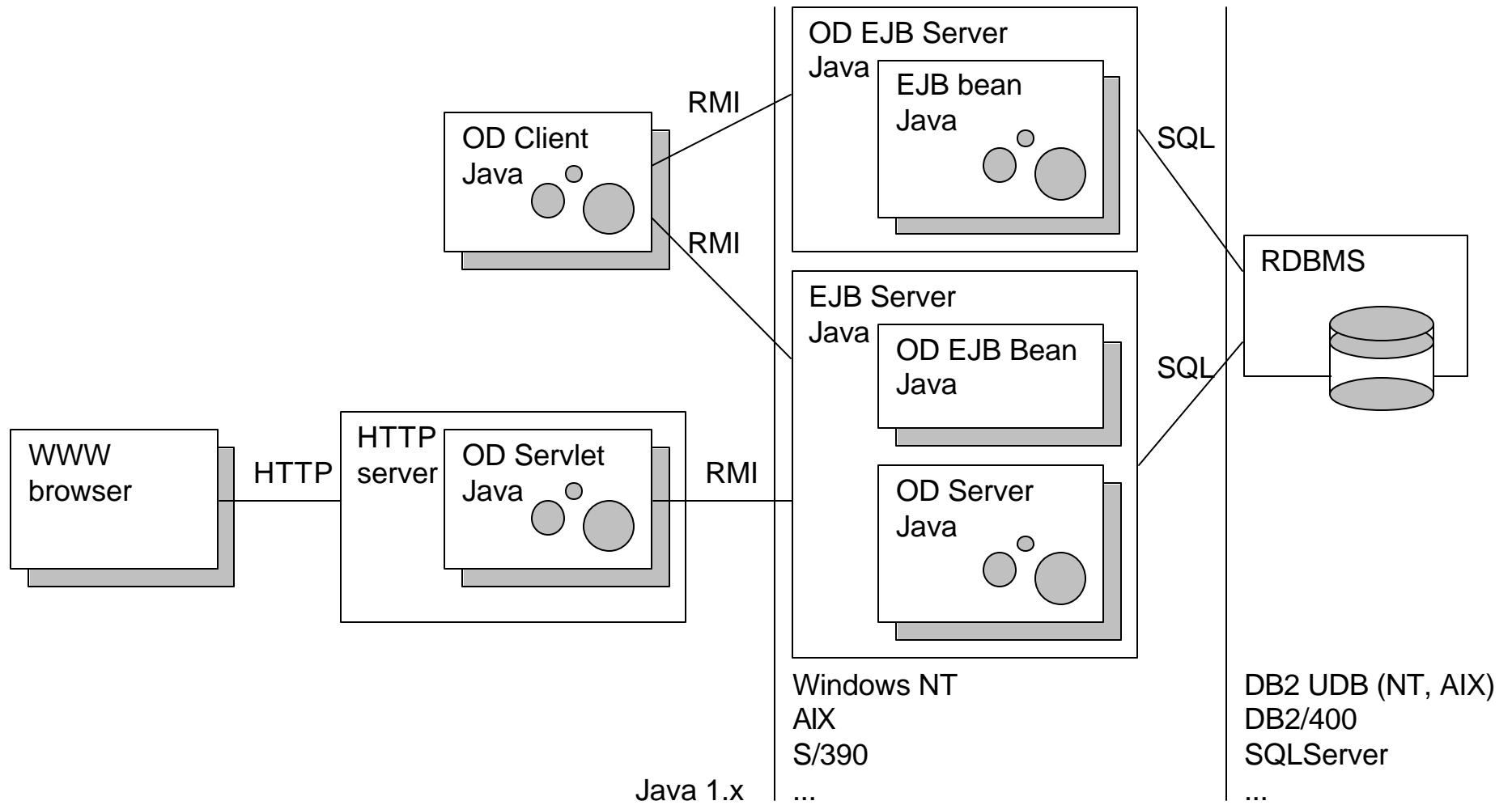


# Once & Done version 2





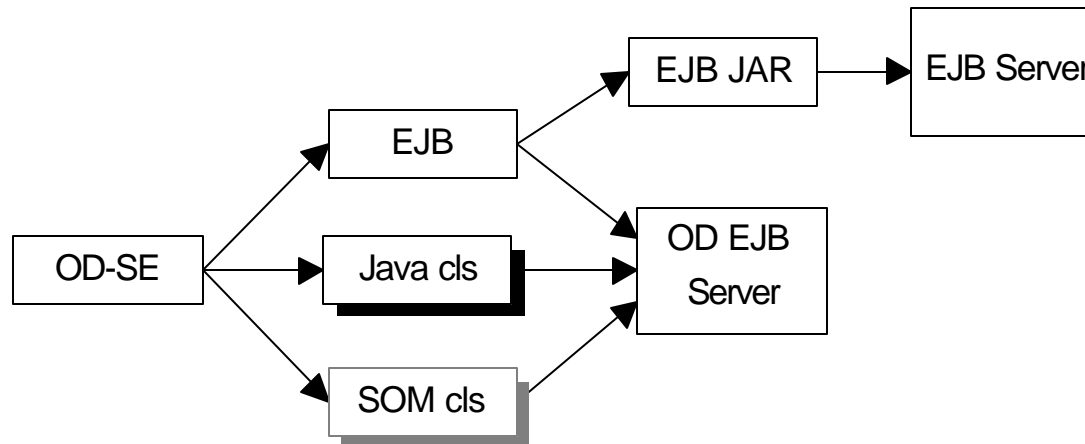
# Once & Done version 3



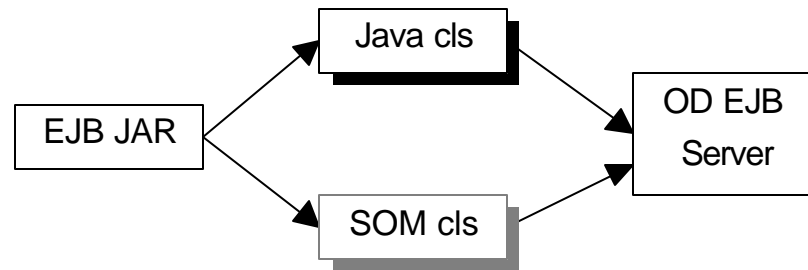


# Enterprise Java Bean'ide kasutamine OD raamistikus

- OD-SE'st genereeritud EJB'de kasutamine:

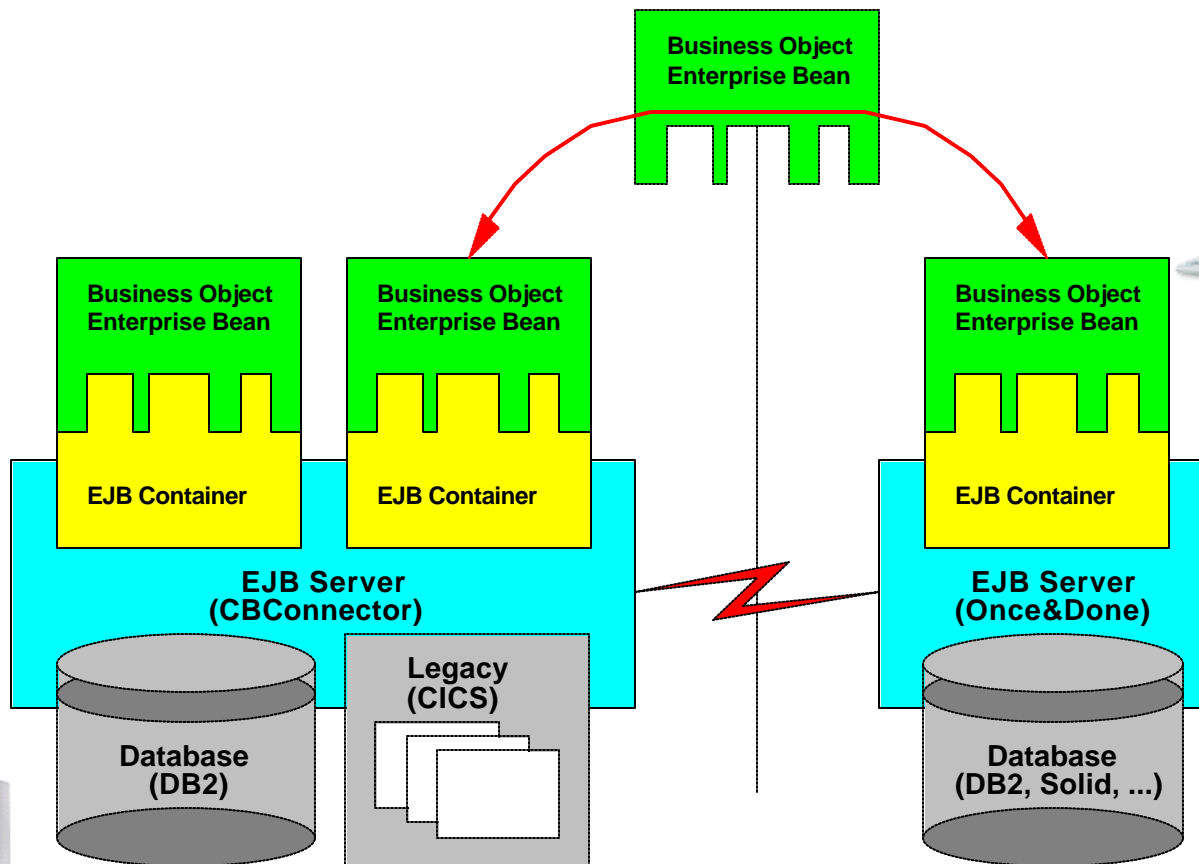


- Kolmanda osapoole tehtud EJB'de kasutamine OD/EJB serveris:





# Äriolemite porditavus OD kasutamisel (Enterprise Java Beans)

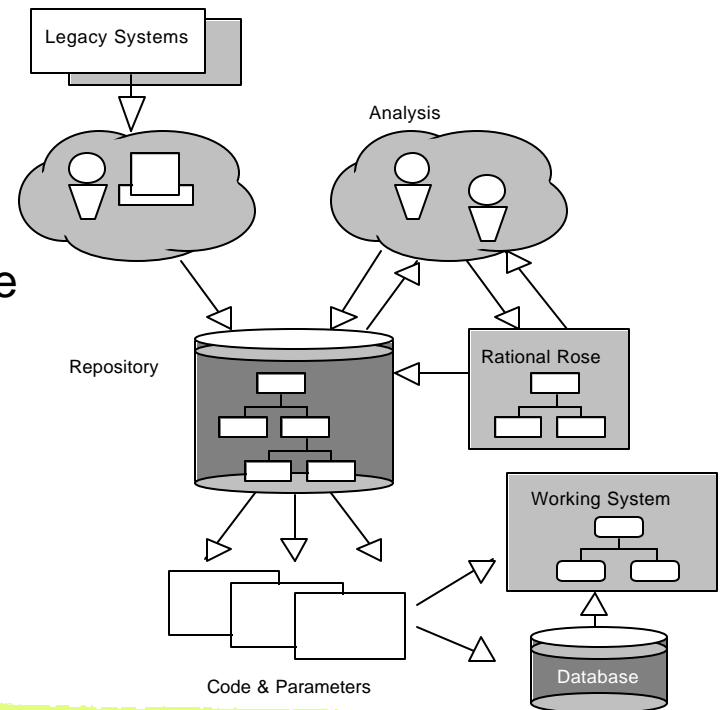




- Once&Done® süsteemide tegemiseks vajalike sammude kirjeldus
- OO paradigmal põhinev iteratiivne tarkvaraprotsess
  - ▶ spetsialiseeritud Once&Done® süsteemide tegemiseks
  - ▶ spetsialiseeritud Once&Done® arhitektuuri ja Once&Done® tööriistade kasutamiseks

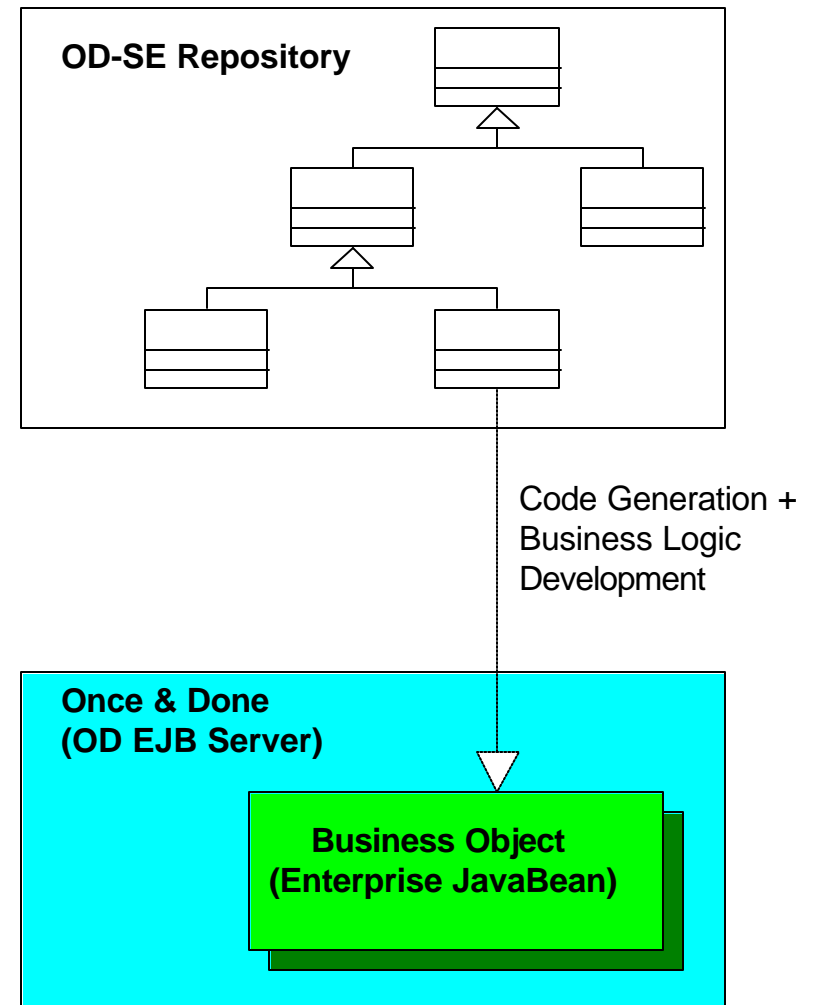
## ■ OD protsessi sammud

- ▶ alustamine (ingl. *inception*)
- ▶ analüüs
  - valdkonna objekt-orienteeritud modelleerimine
  - kindlustustoodete modelleerimine
  - kasutajaliidese modelleerimine
  - prototüüpimine
- ▶ disain
- ▶ realiseerimine
- ▶ lõpetamine



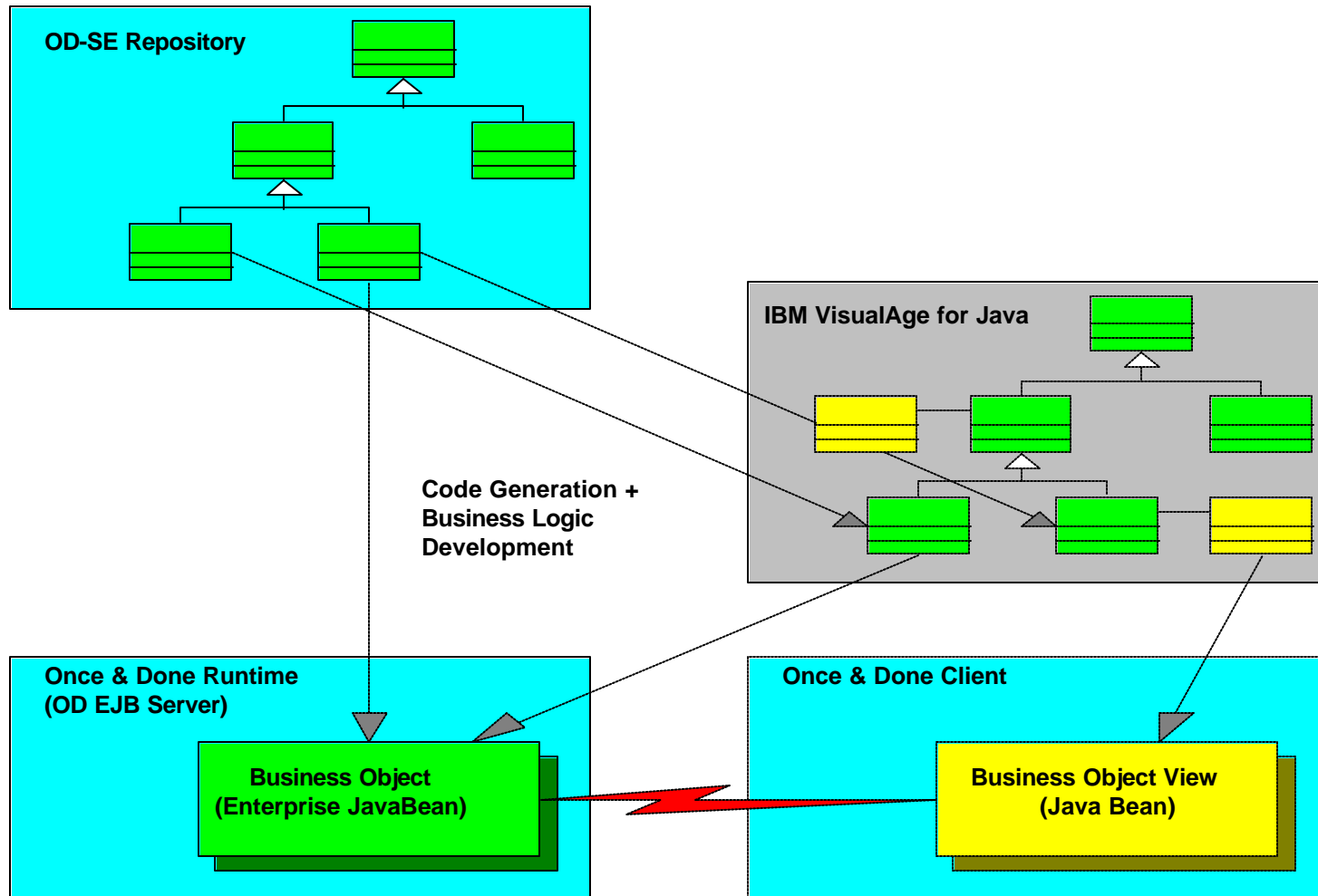
# OD Protsessi sammud

- Alustamine
- Analüüs
  - ▶ ärivaldkonna analüüs
  - ▶ ärivaldkonna objektmodelite loomine
  - ▶ kindlustustoodete modelleerimine
- Disainimudelite loomine
  - ▶ analüüsimudelite täpsustamine
  - ▶ andmebaasiskeemi kirjeldamine
  - ▶ kasutajaliidese kirjeldamine
  - ▶ väljatrükkide kirjeldamine
- Realiseerimine
  - ▶ koodi genereerimine
  - ▶ äriloogika realiseerimine
  - ▶ äriolemite installeerimine OD süsteemi
- Lõpetamine





# Once&Done® süsteemide realiseerimine





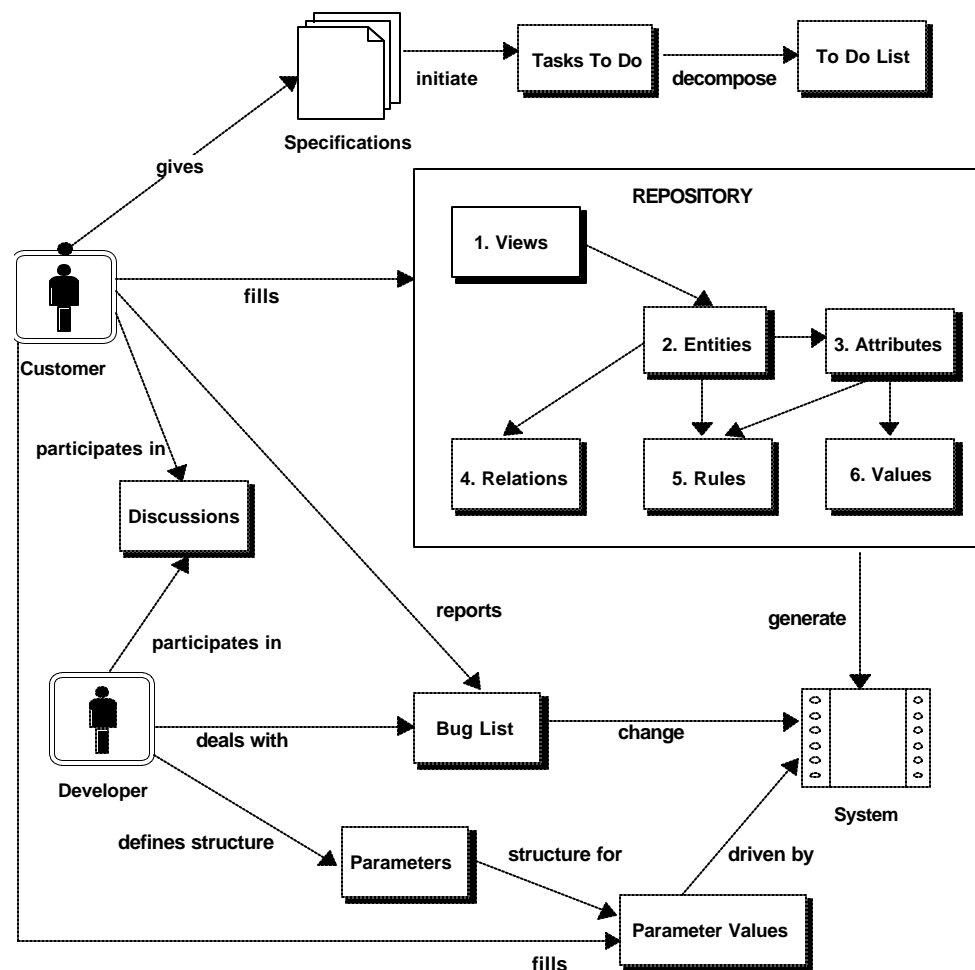
## OD tööriistad

- OD Specification Environment -- OD-SE
  - ▶ OD-SE komponendid
    - OD meta-mudelile vastav andmehoidla (ingl. *repository*)
    - dokumendihalduse rakendused
    - vestlusrakendused
    - projekti halduse rakendused
  - ▶ OD-SE omadused
    - realiseeritud grupivaravahendiga Lotus Notes
    - toetab geograafiliselt hautatud projekttimeeskondi
    - võimaldab ühendada sõltuvusseostega erinevaid andmehoidlaid -- kombineerida OD kindlustusvaldkonna ja teiste Once&Done® süsteemide mudelid
- OD Generaatorid

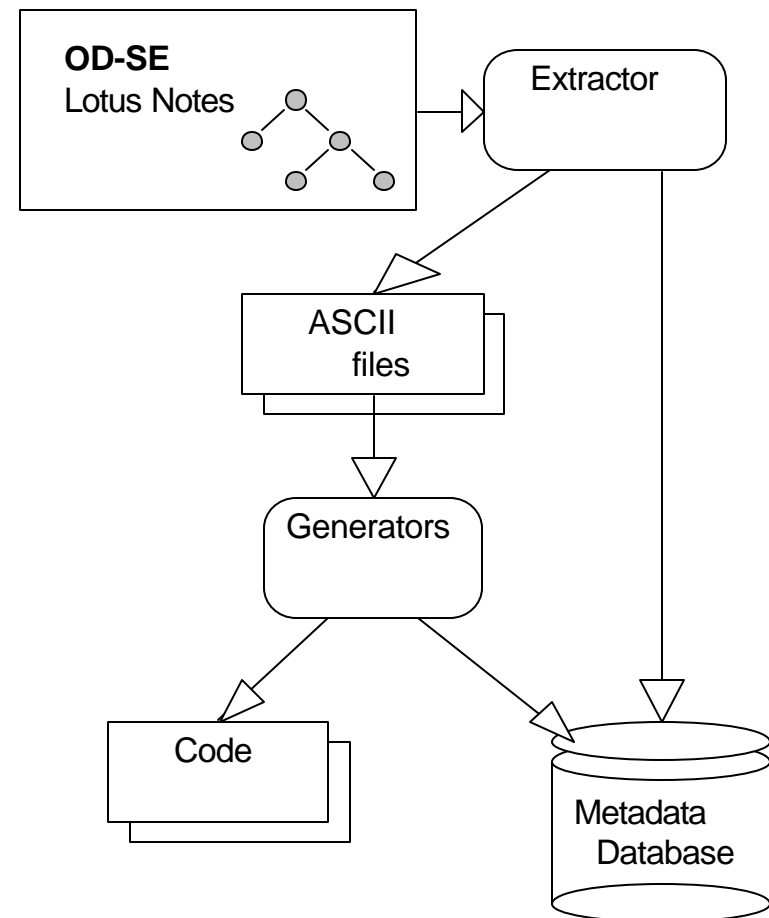
# OD-SE rakendamine OD protsessis

## ■ OD-SE

- ▶ Once&Done® süsteemi spetsifitseerimine
  - 'Specifications' dokumendibaas nõudmistele, spetifikatsioonidele ja testidele
  - 'Repository' baasid
- ▶ OD protsessi toetus
  - 'Discussions' suhtlemiseks
  - 'Tasks To Do' projekti planeerimiseks
  - 'To Do List' projekti juhtimiseks
  - 'Bug List' vigade ja muutussoovide haldamiseks



- Eksporditud failid:
  - ▶ olemite kirjeldused, ärireeglid, andmebaasi kirjeldus, ...
- Olemit kirjeldavad failid:
  - ▶ liidese kirjeldus (.java, .idl, ...)
  - ▶ realisatsioon (.java, .cpp, ...)
  - ▶ lisafailid (makrod, ...)
- Moodulit kirjeldavad failid:
  - ▶ mooduli kirjeldus (.def), *Makefile* (.mak), mooduli initsialiseerimine, ...
- Globaalsed failid:
  - ▶ OD mudelite meta-andmete failid
  - ▶ POS meta-andmete ja juhtfailid
  - ▶ andmebaasi DDL failid
  - ▶ OD töölauda kirjeldavad failid
  - ▶ hinnaarvuts mooduli juhtfailid





# Genererimismalli näide

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE ClassTemplate SYSTEM "class.dtd">
<ClassTemplate Type="Java">

package <PackageName Target="EJBServer"/>;
import javax.ejb.*;
<BasePackages>
import <PackageName Target="EJBServer"/>.*;
</BasePackages>

<Preserve>
// begin of user imports
// end of user imports
</Preserve>

...
```





# Generereerimismalli näide

```
public class <Name/>Bean
    extends <MainParent/>Bean {
<Preserve>
    // ---- Begin of user code
    // ---- End of user code
</Preserve>

<Attributes>
    public <Type/> <Name/>;
</Attributes>
    protected javax.ejb.EntityContext ctx;
<Methods>
    public <Type/> <Name/>(<Args Separator="," ><Type/> <Name/></Args>)
        throws java.rmi.RemoteException {
<Preserve>
    // Begin of code for <ClassName/> method <Name/>
    <MethodDummyReturns/>
    // End of code for <ClassName/> method <Name/>
</Preserve>
    }
</Methods>
    ...
}
```





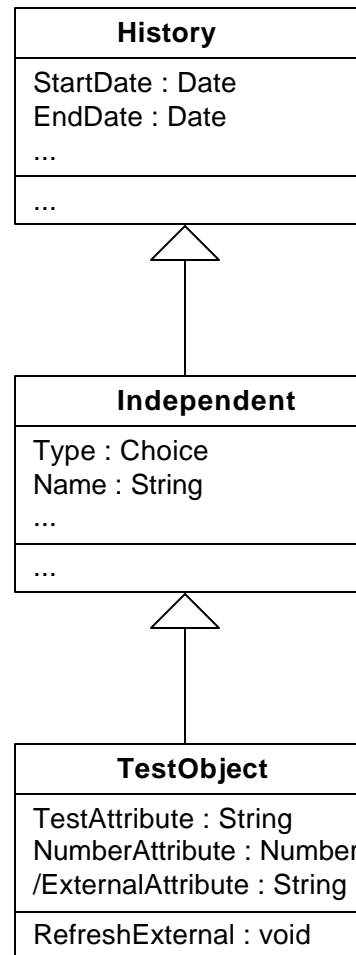
# Genererimismalli näide

```
<Methods Override="Yes">
  public <Type/> <Name/>(<Args Separator=", "><Type/> <Name/></Args>)
    throws java.rmi.RemoteException {
<Preserve>
  // Begin of code for <ClassName/> method <Name/>
  <MethReturnUnlessVoid/> super.<Name/>(<Args Separator=", "><Name/></Args>);
  // End of code for <ClassName/> method <Name/>
</Preserve>
  }
</Methods>

<Methods Inherited="Interface">
  public <Type/> <Name/>(<Args Separator=", "><Type/> <Name/></Args>)
    throws java.rmi.RemoteException {
    <MethReturnUnlessVoid/> super_<OriginalClassName/>.<Name/>(<Args Separator=", "><Name/></Args>);
  }
</Methods>
...
</ClassTemplate>
```



# Repository sisu





# Genereerimitelemuse näide

```
package istest.server.ejb;

import javax.ejb.*;

// begin of user imports
// end of user imports

public class ISDTestObjectBean
    extends ISDIndependentBean {
    // ---- Begin of user code
    // ---- End of user code

    public String extattribute;
    public String testattribute;
    public double numberattribute;
    protected javax.ejb.EntityContext ctx;
    public void CLRefExt()
        throws java.rmi.RemoteException {
        // Begin of code for ISDTestObject method CLRefExt
        // theDO.CLRefExt();
        // End of code for ISDTestObject method CLRefExt
    }
}
...

```

Laienduspunkt



Laienduspunkt



Laienduspunkt

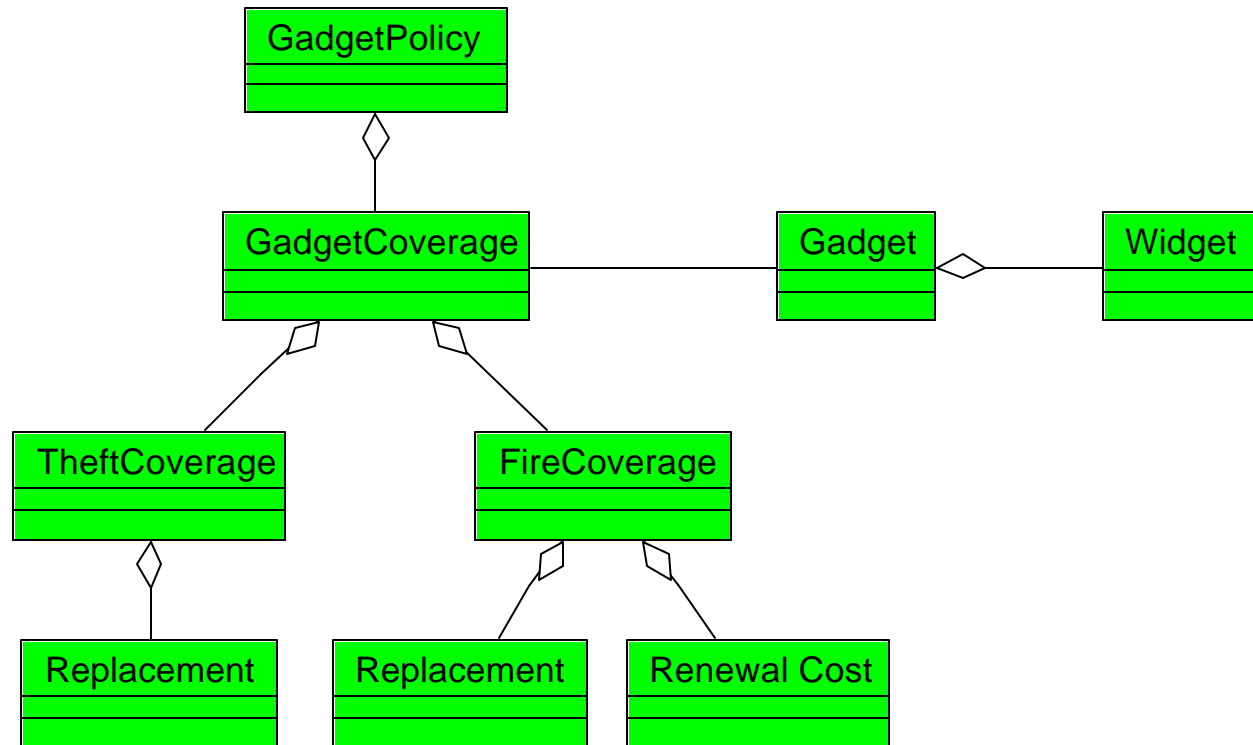




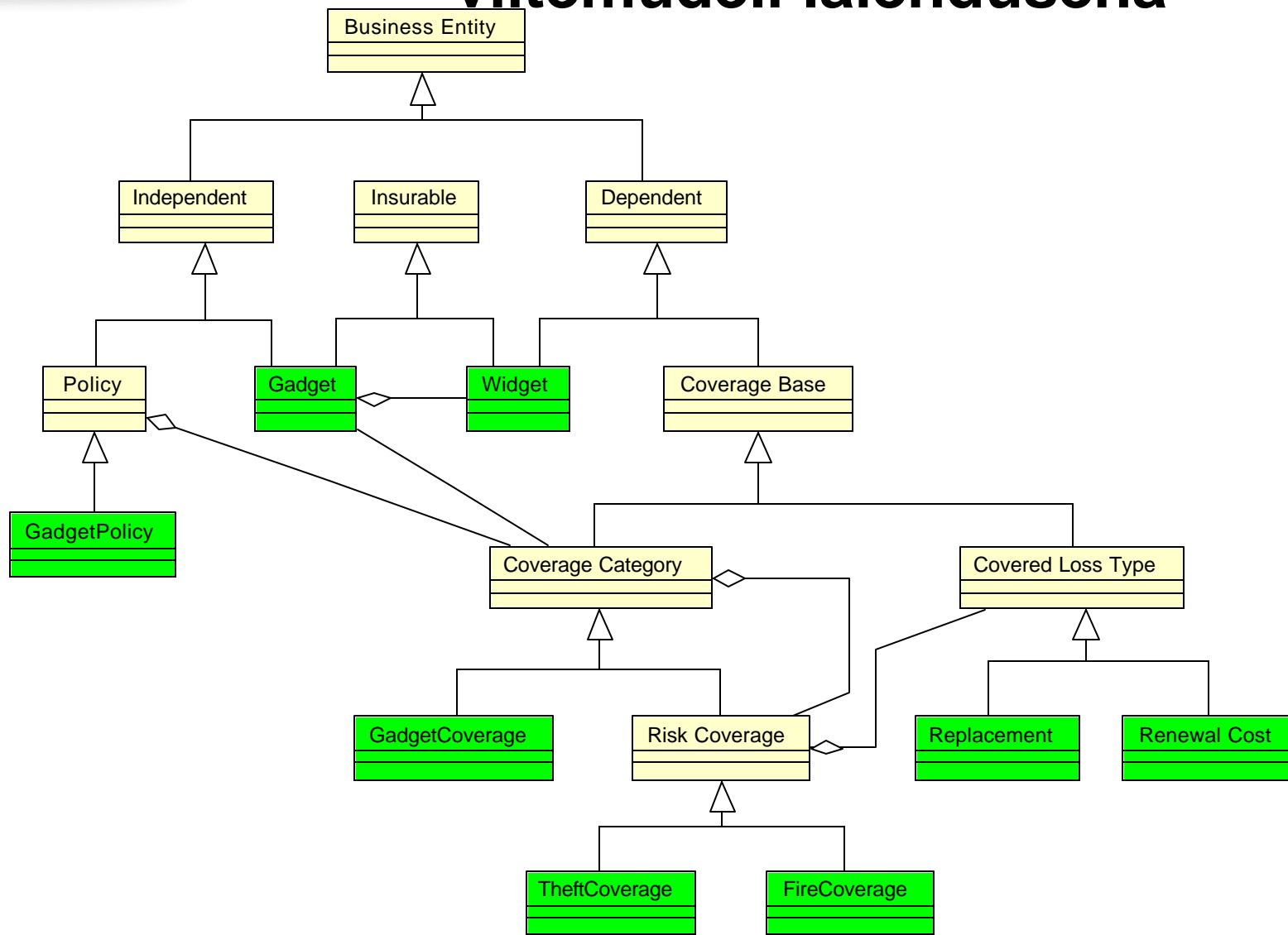
## Once&Done® rakendamise näide

- Värgikindlustus
  - ▶ värk (*gadget*)
  - ▶ vidin (*widget*)
- Värgikindlustuse analüüsimudel
- Värgikindlustuse mudel kindlustusvaldkonna viitemudeli laiendusena
- Värgikindlustuse tootemudel
- Värgikindlustuse disainimudel

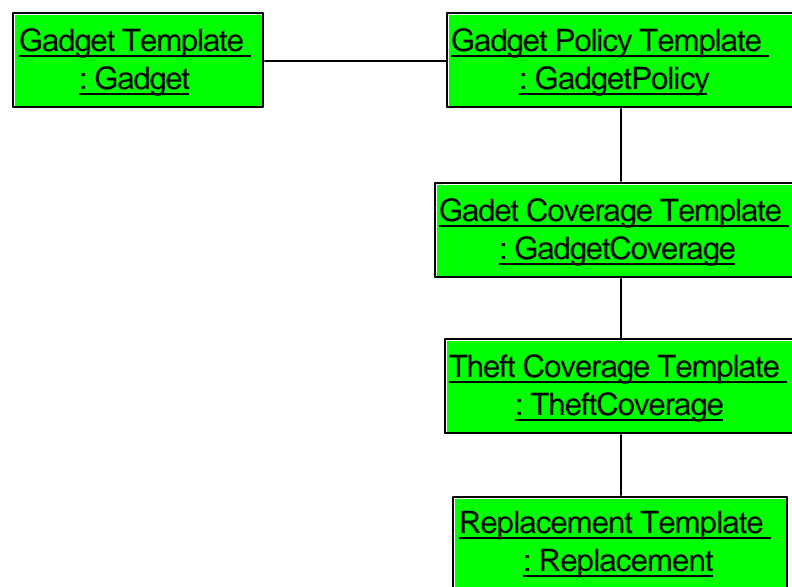
# Näide: Värgikindlustus



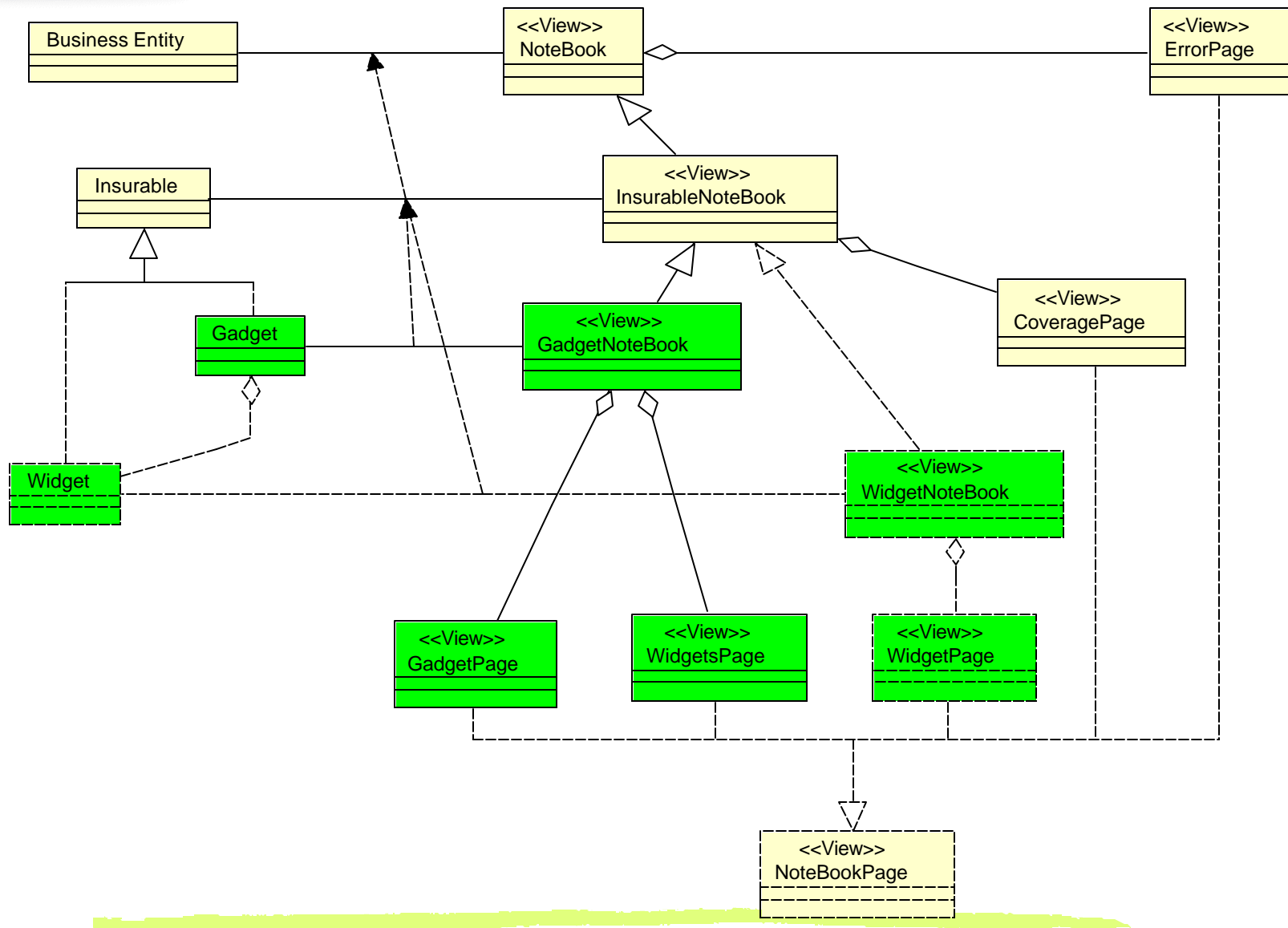
# Näide: Värgikindlustuse mudel viitemudeli laiendusena



# Näide: Värgikindlustuse tootemudel



# Näide: Värgikindlustuse disainimudel







# Once&Done® arengusuunad

## ■ kindlustustegevuse areng

- ▶ keskendumine kliendile
- ▶ täpsem riskide hindamine
- ▶ paindlikumad tooted (näiteks elukindlustuses *unit linked* tooted elukindlustuses)
- ▶ panganduse ja kindlustuse ühtesulamine -- "finantskaubamaja"
- ▶ iseteenendus
- ▶ suurenev infomaht
- ▶ "turuakna" lühenemine

## ■ tehnoloogia areng

- ▶ Internet -- *e-nything* 😊
- ▶ serverikesksed süsteemid, õhukesed (administreerimisvabad) kliendid
- ▶ lai valik erinevaid kliendi- ja meediatüüpe (XML, WAP)
- ▶ standartne, laiendatav, modelleerimiskeel (UML)
- ▶ arendusvahendite integreerimine metainfo alusel (XMI)
- ▶ komponentarhitektuurid (EJB, CCM)
- ▶ platvormist sõltumatu programmeerimiskeel ja API'd (Java)



## Once&Done® rakendamine

- Once&Done® projektid
  - ▶ 3 elukindlustuse poliisihalduse süsteemi
  - ▶ 2 eraisiku kahjukindlustuse poliisihalduse süsteemi
    - neist erinevad versioonid erinevatesse riikidesse
  - ▶ 3 ettevõtte kahjukindlustuse poliisihalduse süsteemi
  - ▶ 1 kahjukäsitluse süsteem
  - ▶ 1 ettevõtte riskianalüüsi süsteem
  - ▶ 1 eraisiku riskianalüüsi süsteem