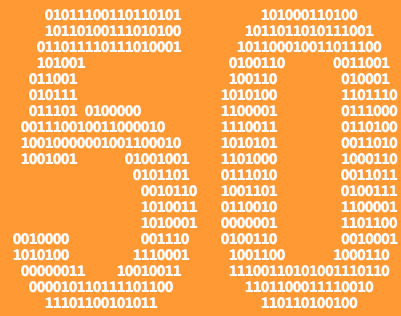
Latvijas Universitātes aģentūra
Latvijas Universitātes Matemātikas un informātikas institūts

Audris Kalniņš

Modeļu transformāciju valoda MOLA un tās lietojumi

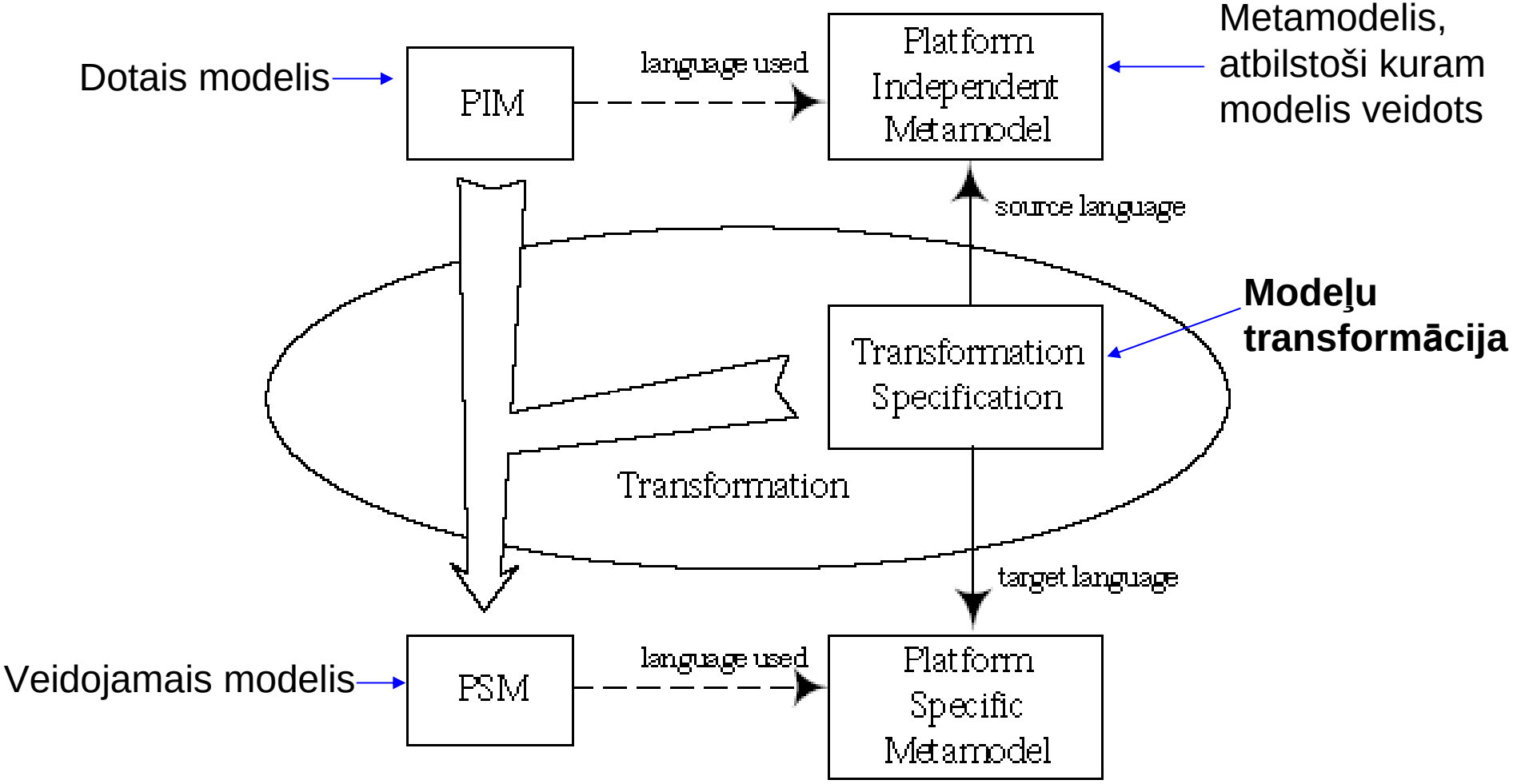


Zinātniskā konference ”Matemātika un informātika pēc 50...”
2009. gada 9. novembris

Kam vajadzīgas modeļu transformācijas

- Sākums - OMG 2001. gadā izsludinātais jaunais lozungs - **MDA** (Model Driven Architecture), vēlāk - **MDSD** (Model Driven Software Development) – **modeļbāzētā izstrāde**.
- Programmatūras izstrāde sākas ar precīzu modeļu virkni kādā modelēšanas valodā – bieži UML, programmas kods ir tikai pēdējais solis
- MDA pieejā tradicionāli ir PIM (platformas neatkarīgais modelis) un PSM (platformai specifiskais modelis)
- MDSD pieejā modeļu izvēle brīvāka
- Ļoti svarīgs uzdevums – kaut daļēji automatizēt pāreju no viena modeļa uz nākamo. To varētu veikt ar **modeļu transformācijām**

• Fragments no OMG skata uz MDA (2003)

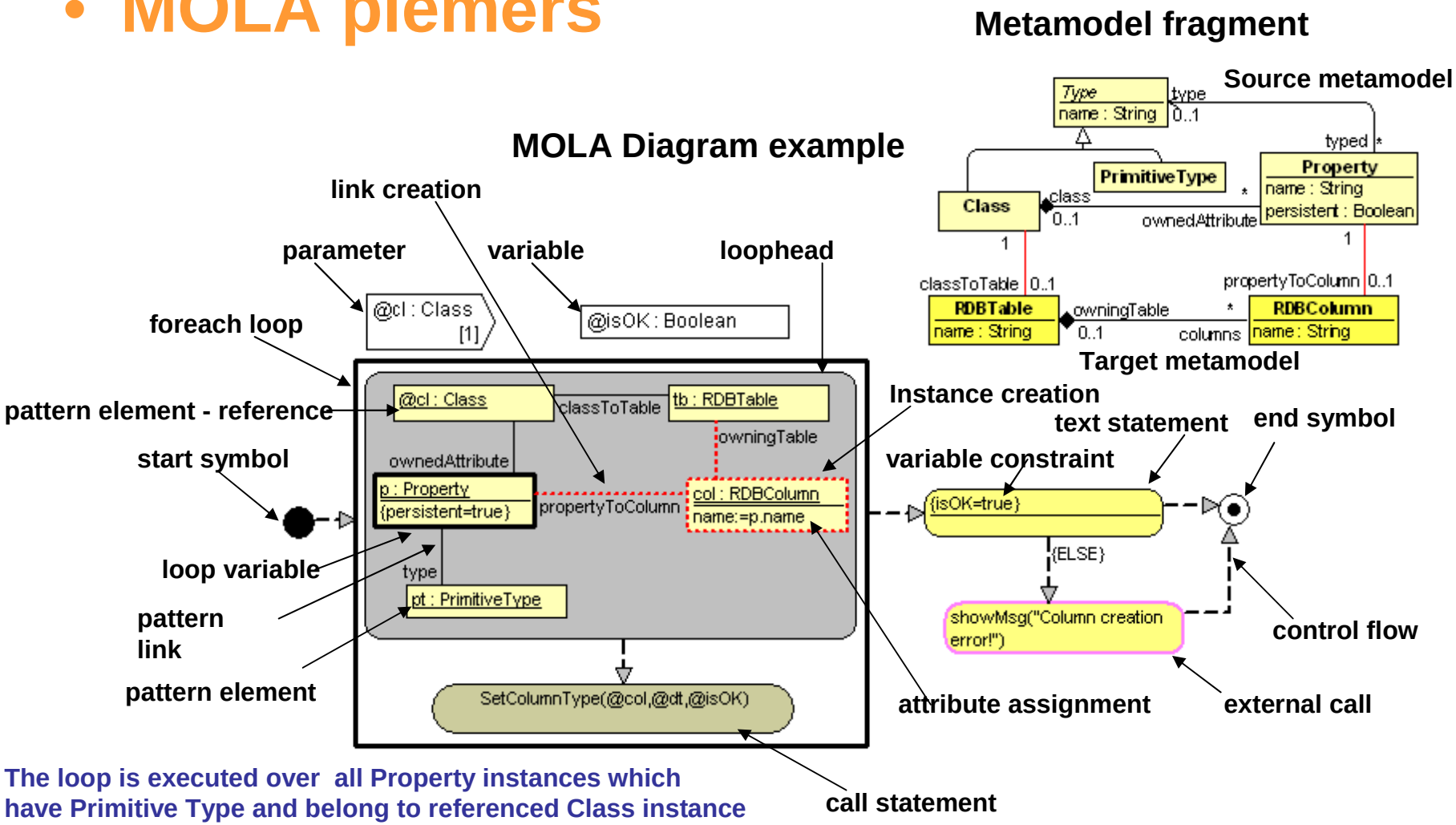


- **Tipiski transformāciju uzdevumi**
  - Kopīgais – **pārveidot vienu modeli par otru**, atbilstoši tam pašam vai citam **metamodelim**, piemēram UML metamodelim
  - Pārveidojums bieži tiek uzdots kā modeļu elementu atbilstība
  - Bieži ir translatora tipa uzdevumi - dotie un iegūstamie modeļi faktiski ir objekti **abstraktajā sintaksē** (klasiskajā MDA pieejā tie ir tieši tādi)
  - Arī rīku būves pasaulē uzdevumi līdzīgi
  - Tipiska situācija – dotajā modelī jāatrod **zināma veida fragments** un atbilstoši tam jāveido cits fragments
  - Klasiskās OOP valodas (Java u.c.) šiem uzdevumiem nav labākās

- **Risinājums – transformāciju valodas**
  - Sākums - 2002. g. aprīlī OMG izsludināja konkursu uz transformāciju valodas standartu (Request for Proposal: MOF 2.0 Query / Views / Transformations)
  - Pirmais MOF QVT standarts tikai 2008 gadā
  - Pa to laiku citas transformāciju valodas ieguva popularitāti – ATL, GReAT, Viatra (visas ~2003.g)
  - Faktiski šādas valodas bija daudz agrāk – grafu transformāciju valodas (PROGRES – 1996, AGG – 2001), tikai nebija plaša pielietojuma
  - 2003. g beigās LU MII sākts darbs pie valodas MOLA izveides

- # Transformāciju valoda MOLA
  - Pirmās publikācijas **2004**. g – A.Kalniņš, J.Bārzdiņš, E.Celms, uzreiz ieguva atpazīstamību pasaulē
  - **Grafiska valoda**, kas orientēta uz vieglu transformāciju veidošanu un lasāmību
  - Balstās uz tipisko konstrukciju "**fragmenta atrašana**" (*pattern matching*), bet izpildes vadībai izmanto tradicionālās strukturētas programmēšanas idejas
  - Par valodu MOLA ir 10 publikācijas starptautiskos izdevumos, 1 aizstāvēta un 1 sagatavota doktora disertācija

• MOLA piemērs



The loop is executed over all Property instances which have Primitive Type and belong to referenced Class instance if it is already mapped to an RDBTable

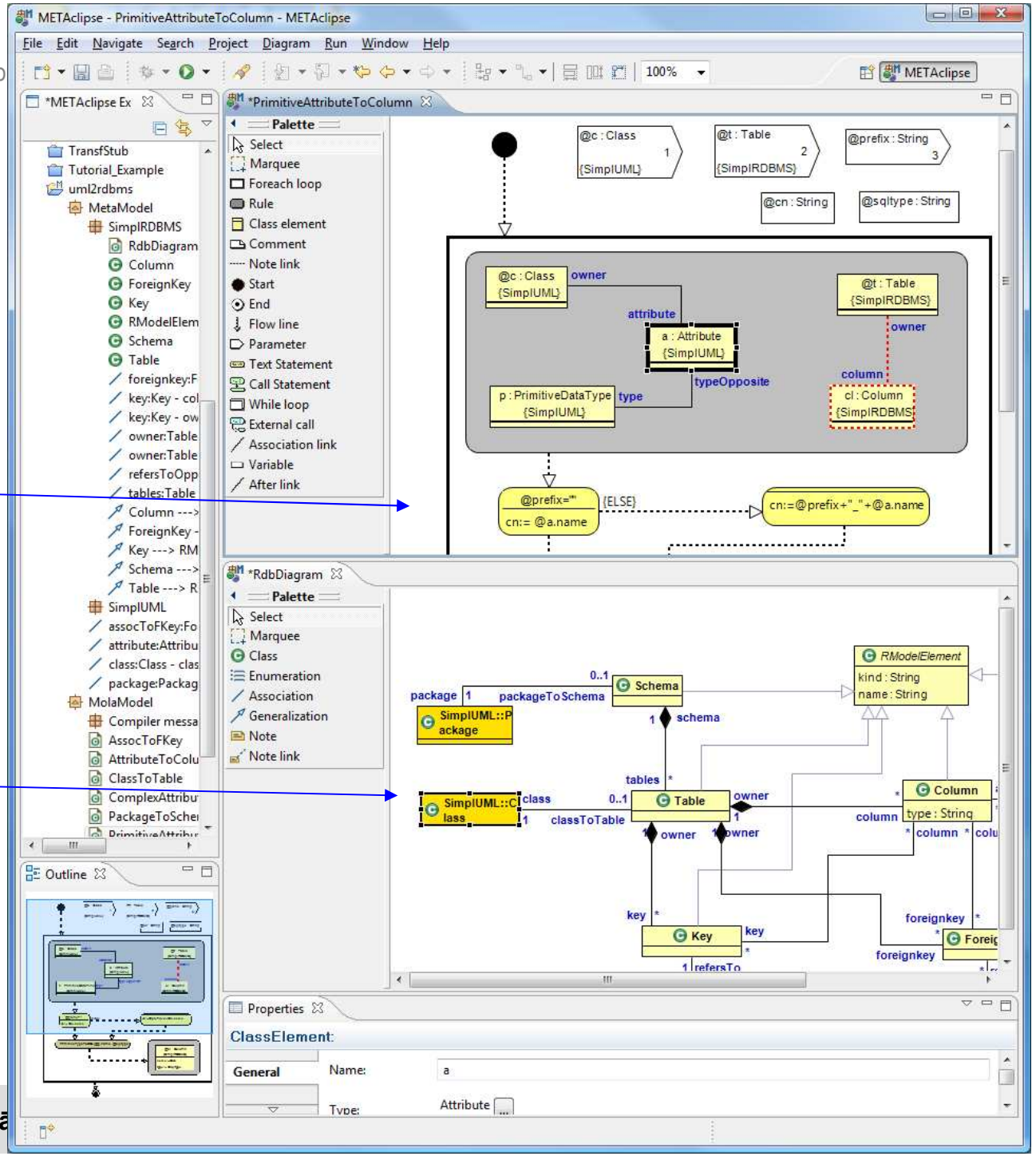
- **Valodas MOLA atbalsta rīki**
  - MOLA atbalsta rīki sastāv no **grafiskajiem redaktoriem**, **kompilatora** un izpildes nodrošinājuma
  - Izstrādātas vairākas versijas šiem rīkiem
  - Pašreizējā versija jau lielā mērā rūpnieciska realizācija
  - Uz šīs versijas pamata veikti nopietni MOLA lietojumi
  - Atbalsta rīki brīvi pieejami lejupielādei, tie veidoti kā brīvi pieejamās Eclipse vides spraudņi

- **Grafiskie redaktori MOLA rīkam**
  - Sākotnējā versija – LU MII rīku būves vidē GME (EBM) – 2004.g
  - Īstā versija – LU MII metamodeļu/transformāciju balstītajā rīku būves vidē **METAclipse** – 2007.g, rūpniecisks variants – 2008. g - Elīna Kalniņa
  - Nodrošina visas lietotāja ērtības grafiskai programmēšanas valodai – konteksta atkarīgi "priekšā teicēji", vārdu konsistences atbalsts, ...
  - Funkcionalitāti nodrošina transformācijas, kas definētas tai pašā valodā MOLA – tipisks *bootstraping* lietojums

• Grafiskie redaktori - piemērs

MOLA diagramma (MOLA procedūra)

Klašu diagramma (metamodeļa fragments)

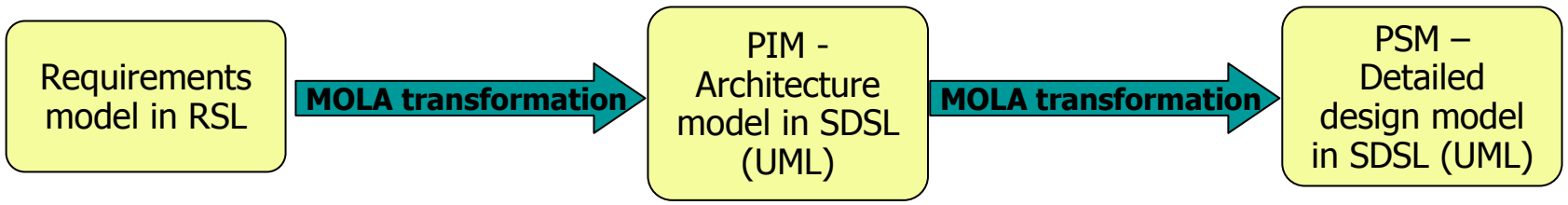


- **MOLA kompilators**
  - Galvenais realizācijas autors – Agris Šostaks
  - Pirmā MOLA realizācija – izmantojot interpretatoru un SQL datu bāzi kā modeļu repozitoriju (2005.g.)
  - Īstā realizācija – **optimizējošs kompilators** uz zemāka līmeņa transformācijas valodu (**L3**, **L0**)
  - Tālāk notiek kompilācija uz C++ vai Java
  - Izmantojamās modeļu **repozitorijas** – miirep (LU MII), EMF (Eclipse standarts), JGraLab (UKo)
  - Būtiskākā ideja – optimāla *pattern matching* realizācija, kas arī nosaka visu transformācijas ātrdarbību
  - Nodrošināti pamatlīdzekļi transformāciju testēšanai

- # Galvenie MOLA lietojumi

1- Eiropas 6.FP IST projektā ReDSeeDS

- Visa MDSD pieeja projektā (prasību modelis -> PIM -> PSM) balstās uz netriviālām transformācijām valodā MOLA
- Rezultātā no formalizētām prasībām var automātiski izveidot "gandrīz izpildāmu" sistēmas prototipu
- Kopējais transformāciju apjoms – 220 procedūras (vienam stilam)
- Norealizēts imports un eksports uz UML rīku Enterprise Architect (arī ar MOLA transformāciju palīdzību), rezultātā transformētos modeļus var skatīt un tālāk modificēt UML rīkā
- Parāda MOLA lietojamību rūpnieciskas transformāciju vides lomā



• Galvenie MOLA lietojumi

2 – **Rīku būvē – METAclipse platformā**

- METAclipse ir transformāciju bāzēta rīku būves platforma, izstrādāta LU MII (galvenais autors Oskars Vilītis)
- Konkrētā rīka funkcionalitāti definē ar transformācijām MOLA valodā
- Pamatdarbības veic izpildes dziņi (*engines*), balstoties uz fiksētu metamodeli
- Īpaši piemērota vide sarežģītu grafisku DSL (piemēram, transformāciju valodu) realizācijai
- Pašreizējais MOLA redaktors realizēts METAclipse vidē
- Ievērojams apjoms - ~ 440 procedūru, tomēr realizācija izrādījās relatīvi viegla
- Veiktspēja pietiekami laba

- **Turpmākās perspektīvas**
  - Paredzēti nelieli MOLA valodas papildinājumi
  - Galvenais attīstības virziens – nodrošināt vieglāku transformāciju uzdošanu, tai skatā ar atbilstībām (*mappings*)
  - Tādēļ nepieciešamas arī augstāka līmeņa transformācijas (**HOT**), ar kurām veido pašas transformācijas
  - Šim noliekam tiek veidota jauna veida valoda – **Template MOLA**, kas ir grafiska šablonu valoda
  - Rezultātā būs iespējama arī jauna pieeja rīku būvei

- **Secinājumi**
  - Valodas MOLA un tās rīku izstrāde izrādījusies ļoti veiksmīgs projekts, kad izstrādātāju kolektīva kvalifikācija (modelēšana, metamodelēšana, rīku būve) ļoti atbilda populāram uzdevumam
  - Svarīgi bija laikā pamanīt uzdevumu (OMG u.c. avoti)
  - Transformāciju valodu pētījumi kļuva par pamatu tālākiem darbiem rīku būvē, ontoloģijās u.c.

Paldies par uzmanību!