**RĪGAS TEHNISKĀ UNIVERSITĀTE**

Datorzinātnes un informācijas tehnoloģijas fakultāte

Lietišķo datorsistēmu institūts

Praktisko darbu pārskats mācību priekšmetā

**MODEĻVADĀMAS PROGRAMMATURĀS IZSTRĀDES PRAKTIKUMS**

Izstrādāja: Sergejs Terentjevs

2. kurss, 1. grupa

apl. nr.: 061RDB140

Pārbaudīja: Doc. Ē.Asņina

Rīga, 2015

Saturs

1. praktiskais uzdevums 3

1.1. Uzdevuma nostādne 3

1.2. Risinājums 3

2. praktiskais uzdevums 9

2.1. Uzdevuma nostādne 9

2.2. Risinājums 9

3. praktiskais uzdevums 11

3.1. Uzdevuma nostādne 11

3.2. Risinājums 11

4. praktiskais uzdevums 16

4.1. Uzdevuma nostādne 16

4.2. Risinājums 16

5. praktiskais uzdevums 24

5.1. Uzdevuma nostādne 24

5.2. Risinājums 24

6. praktiskais uzdevums 28

6.1. Uzdevuma nostādne 28

6.2. Risinājums 28

Secinājumi 33

# praktiskais uzdevums

## Uzdevuma nostādne

1. Izdomāt un aprakstīt brīvā formā programmatūras sistēmas funkcionalitāti. Apraksta apjoms viena rindkopa (5-8 rindas).
2. Aprakstīt izdomātās sistēmas klašu kopu CRC (Class-Responsibility-Collaborator) karšu veidā.

## Risinājums

Uzdevuma ietvaros tiek projektēta biroja preču iegādes sistēma. Projektējums ietver sevī vairākas pētāmos objektus:

* Darbinieks – sistēmas lietotājs, kas veic preces pasūtīšanu, vadoties pēc pieejamam precēm katalogā;
* Nodaļas vadītājs – sistēmas lietotājs, kurš ir pilntiesīgs gan akceptēt darbinieku pasūtījumu, gan arī pats veikt preces pasūtīšanu;
* Preču katalogs – paredz operācijas preču izvadei un modifikācijai;
* Prece – priekšmets, kurš ir pieejams iegādei;
* Preces vienība – pasūtījuma sastāvdaļa, kurā tiek ietvērtā gan prece, gan arī tās daudzums. Palīdz sarēķināt pasūtījuma kopējas izmaksas, ļaujot noteikt cenu bez PVN, ka arī paša PVN vērtību.
* Pasūtījums – lietotāja darbības rezultāts, kurām piemīt preču iegādes status un nepieciešamas darbības pasutījuma pārvaldībai.

Projektējot klases visiem atribūtiem ir definēta slēgta (*private*) pieeja, kamēr klases metodēm ir definēta publiska (*public*) piekļuve.

Darba ietvaros tiek definēti pārskaitījumi, lai definētu preču kategorijas, darbinieku nodaļas un pasutījumu statusu, t.i., vai ir pasūtīts, akceptēts, atcelts vai saņemts.

1. tabula

Klase *Darbinieks*

|  |  |
| --- | --- |
| Klases vārds: *Darbinieks* | |
| Klases tips: *loma* | |
| Klases raksturojumi: *taustāma, nedalāma, secīga, pastāvoša, piesardzīga* | |
| Atbildības:  ***atribūti:***  *vards*  *uzvards*  *personasKods*  *nodala*  *vaditajs*  *pasutijumi*  ***metodes:***  *iegutVardu*  *pieskirtVardu*  *iegutUzvardu pieskirtUzvardu iegutPersonasKodu*  *pieskirtPersonasKodu iegutNodalu pieskirtNodalu iegutNodalasVaditaju pieskirtNodalasVaditaju iegutPasutijumus pieskirtPasutijumus parceltUzNodalu atlaist labot* | Sadarbības klases:  *Nodala*  *NodalasVaditajs*  *Pasutijums*  *Nodala*  *Nodala*  *NodalasVaditajs*  *NodalasVaditajs*  *Pasutijums*  *Pasutijums*  *Nodala* |

2. tabula

Klase *Nodaļas vadītājs*

|  |  |
| --- | --- |
| Klases vārds: *NodalasVaditajs* | |
| Klases tips: *loma* | |
| Klases raksturojumi: *taustāma, nedalāma, secīga, pastāvoša, piesardzīga* | |
| Atbildības:  ***atribūti:***  *vards*  *uzvards*  *personasKods*  *nodala*  *pasutijumi darbinieki*  ***metodes:***  *iegutVardu*  *pieskirtVardu*  *iegutUzvardu pieskirtUzvardu iegutPersonasKodu*  *pieskirtPersonasKodu*  *iegutNodalu pieskirtNodalu iegutDarbiniekus pieskirtDarbinieks iegutPasutijumus pieskirtPasutijumus pievienotDarbinieku iegutAkceptetosPasutijumus iegutPasutijumusAkceptam* | Sadarbības klases:  *Nodala*  *Pasutijums*  *Darbinieks*  *Nodala*  *Nodala*  *Darbinieks*  *Darbinieks*  *Pasutijums*  *Pasutijums*  *Darbinieks*  *Pasutijums*  *Pasutijums* |

3. tabula

Klase *Preču katalogs*

|  |  |
| --- | --- |
| Klases vārds: *PrecuKatalogs* | |
| Klases tips: *struktūras vienums* | |
| Klases raksturojumi: *abstrakta, agregāts, secīga, pārejoša, piesardzīga* | |
| Atbildības:  ***atribūti:***  *preces*  ***metodes:***  *pieskirtPreces*  *iegutPreces*  *atlasit*  *pievienotPreci dzestPreci samekletPreci* | Sadarbības klases:  *Prece*  *Prece*  *Prece*  *Kategorija, Prece  Prece Prece Prece* |

4. tabula

Klase *Prece*

|  |  |
| --- | --- |
| Klases vārds: *Prece* | |
| Klases tips: *cita „lieta”* | |
| Klases raksturojumi: *taustāma, nedalāma, secīga, pastāvoša, piesardzīga* | |
| Atbildības:  ***atribūti:***  *nosaukums*  *kategorija*  *vienibasCena razotajaKods*  *vieniba*  ***metodes:***  *iegutNosaukumu*  *iegutKategoriju iegutVienibasCenu iegutRazotajaKodu*  *iegutVienibu pieskirtNosaukumu pieskirtKategoriju pieskirtVienibasCenu pieskirtRazotajaKodu*  *pieskirtVienibu*  *atjaunot* | Sadarbības klases:  *Kategorija*  *PrecesVieniba*  *Kategorija*  *PrecesVieniba  Kategorija   PrecesVieniba* |

5. tabula

Klase *Preces vienība*

|  |  |
| --- | --- |
| Klases vārds: *PrecesVieniba* | |
| Klases tips: *cita „lieta”* | |
| Klases raksturojumi: *taustāma, agregāts, secīga, pastāvoša, piesardzīga* | |
| Atbildības:  ***atribūti:***  *daudzums*  *prece pasutijumi*  ***metodes:***  *iegutDaudzumu pieskirtDaudzumu*  *iegutPasutijumu pieskirtPasutijumu iegutPreci pieskirtPreci sarekinatKopejoCenu noteiktCenuBezPVN noteiktPVN* | Sadarbības klases:  *Prece*  *Pasutijums*  *Pasutijums*  *Pasutijums*  *Prece* *Prece* |

6. tabula

Klase *Pasūtījums*

|  |  |
| --- | --- |
| Klases vārds: *Pasutijums* | |
| Klases tips: *cita „lieta”* | |
| Klases raksturojumi: *taustāma, agregāts, secīga, pastāvoša, piesardzīga* | |
| Atbildības:  ***atribūti:***  *numurs*  *status*  *pasutisanasDatums akceptesanasDatums sanemsanasDatums*  *darbinieks*  *vienibas*  ***metodes:***  *iegutNumuru iegutStatusu*  *iegutPasutisanasDatumu iegutAkceptesanasDatumu*  *iegutSanemsanasDatumu iegutDarbinieku pieskirtDarbinieku iegutPrecesVienibu pieskirtPrecesVienibu akceptet pasutit sanemt atcelt* | Sadarbības klases:  *PasutijumaStatus*  *Darbinieks*  *PrecesVieniba*    *PasutijumaStatus*  *Darbinieks Darbinieks*  *PrecesVieniba*  *PrecesVieniba* |

7. tabula

Klase *Nodaļa*

|  |  |
| --- | --- |
| Klases vārds: *Nodala* | |
| Klases tips: *struktūras vienums* | |
| Klases raksturojumi: *abstrakta, nedalāma, secīga, pārejoša, sabojājama* | |
| Atbildības:  ***atribūti:***  *ADMINISTRATIVA*  *TIRZNIECIBAS RAZOSANAS PARVALDIBAS* | Sadarbības klases: |

8. tabula

Klase *Kategorija*

|  |  |
| --- | --- |
| Klases vārds: *Kategorija* | |
| Klases tips: *struktūras vienums* | |
| Klases raksturojumi: *abstrakta, nedalāma, secīga, pārejoša, sabojājama* | |
| Atbildības:  ***atribūti:***  *AKSESUARI*  *BIROJA\_PRECES GALDA\_PIEDERUMI RAKSTAMPIEDERUMI* | Sadarbības klases: |

9. tabula

Klase *Pasūtījuma Status*

|  |  |
| --- | --- |
| Klases vārds: *PasutijumaStatus* | |
| Klases tips: *struktūras vienums* | |
| Klases raksturojumi: *abstrakta, nedalāma, secīga, pārejoša, sabojājama* | |
| Atbildības:  ***atribūti:***  *PASUTITS*  *AKCEPTETS SANEMTS ATCELTS* | Sadarbības klases: |

# praktiskais uzdevums



## Uzdevuma nostādne

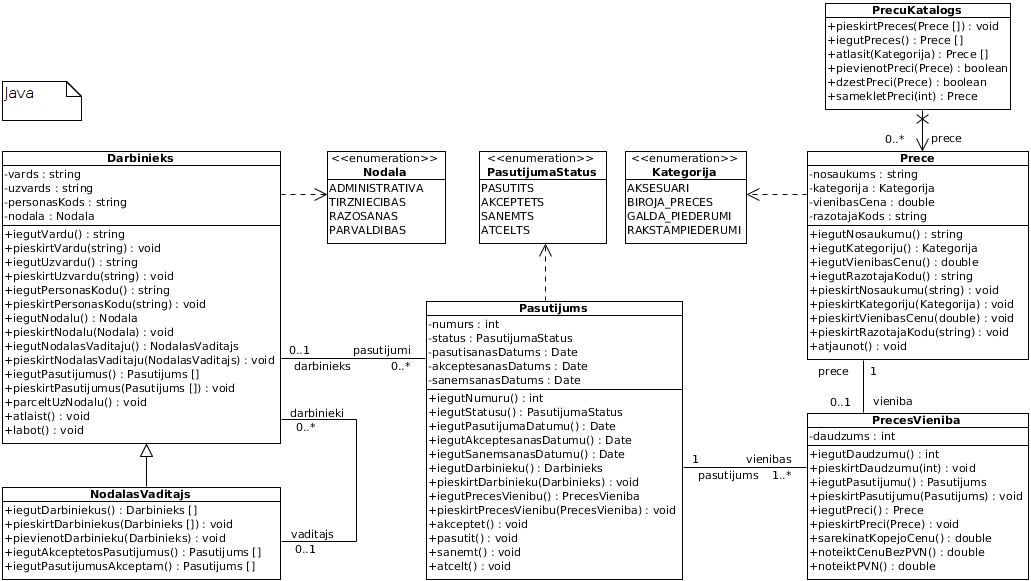
Izveidot UML klašu diagrammu atbilstoši 1. praktiskā uzdevumā definētām un pilnveidotām klasēm CRC veidā.

*Diagrammas līmenis ir platformai specifisks (PSM)!* Platforma uzdevuma ietvaros ir viena no programmēšanas valodām Java, C#.NET vai C++. Lūdzu diagrammā elementā *Note (piezīme)* norādīt izmantojamo programmēšanas valodu.

Klašu diagrammā jābūt vismaz 5 klasēm, kuru sadarbība ar citām klasēm ir attēlota, izmantojot atbilstošu attiecību veidus.

## Risinājums

Izveidota UML klašu diagramma ir attēlota 2.1 attēlā. Par platformas valodu ir izvēlēta Java programmēšanas valoda. Diagrammā ir pievienoti pārskaitījuma (*enum*) datu struktūras.



* 1. att. Biroja preču pasūtīšanas sistēmas UML klašu diagramma.

# praktiskais uzdevums

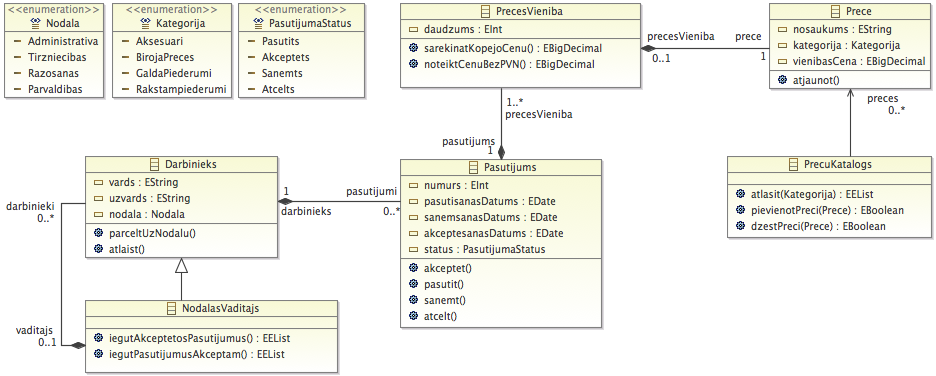


## Uzdevuma nostādne

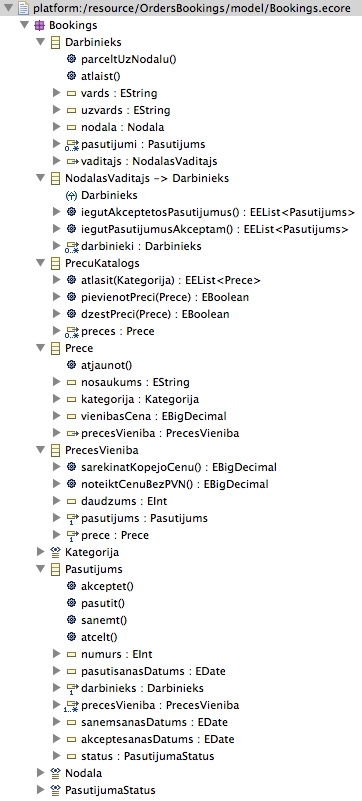
1. Uzstādīt Eclipse ar "ATL SDK - ATLAS Transformation Language SDK"
2. Uzstādīt no "Eclipse Indigo R Packages" rīkus "Eclipse Modeling Tools". URL: [http://www.eclipse.org/downloads/packages/release/indigo/r](http://www.eclipse.org/downloads/packages/release/indigo/r" \t "_blank) un pēc tam uzstādīt ATL spraudni - kā to izdarīt ir aprakstīts šeit [http://wiki.eclipse.org/ATL/Tutorials\_-\_Create\_a\_simple\_ATL\_transformation](http://wiki.eclipse.org/ATL/Tutorials_-_Create_a_simple_ATL_transformation" \t "_blank)
3. vai lējupielādēt Eclipse ar jau pievienoto spraudni ATL transformācijām no [http://www.eclipse.org/atl/download/](http://www.eclipse.org/atl/download/" \t "_blank)
4. Izveidot tukšu ATL projektu.
5. Izveidot Ecore modeli (<http://www.vogella.de/articles/EclipseEMF/article.html>) savai UML klašu diagrammai (piemēru skatieties mācību materiālā [http://wiki.eclipse.org/ATL/Tutorials\_-\_Create\_a\_simple\_ATL\_transformation](http://wiki.eclipse.org/ATL/Tutorials_-_Create_a_simple_ATL_transformation" \t "_blank) (sadaļas The metamodels, The Families metamodel)). Tas būs Jūsu ieejas modelis transformācijām, otro failu varat arī izveidot, bet atstāt tukšu. *Parametrus operācijām norādīt nevajag, tikai atgriežamas vērtības tipu. Katrai klasei jāievada NE VAIRĀK par divām operācijām un trīs atribūtiem. Attiecībām starp klasēm jābūt visām. Tipi ir iekšēji Ecore tipi.*

## Risinājums

Veicot uzdevumu tika izveidota 3.1 attēlā attēlota *ECore* diagramma. Uz tas pamatā tika izstrādāts 3.2 attēlā redzamais *Ecore* modelis.



* 1. att. Biroja preču pasūtīšanas sistēmas *ECore* diagramma.



* 1. att. Izveidotais *ECore* modelis.

*Ecore* modeļa XML shēma izskatās sekojoši:

<?xml version="1.0" encoding="UTF-8"?>

<ecore:EPackage xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore" name="Bookings">

<eClassifiers xsi:type="ecore:EClass" name="Darbinieks">

<eOperations name="parceltUzNodalu"/>

<eOperations name="atlaist"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="vards"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EString"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="uzvards"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EString"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="nodala"

eType="#//Nodala"/>

<eStructuralFeatures xsi:type="ecore:EReference" name="pasutijumi"

upperBound="-1"eType="#//Pasutijums" containment="true"

eOpposite="#//Pasutijums/darbinieks"/>

<eStructuralFeatures xsi:type="ecore:EReference" name="vaditajs"

eType="#//NodalasVaditajs" eOpposite="#//NodalasVaditajs/darbinieki"/>

</eClassifiers>

<eClassifiers xsi:type="ecore:EClass" name="NodalasVaditajs"

eSuperTypes="#//Darbinieks">

<eOperations name="iegutAkceptetosPasutijumus">

<eGenericType eClassifier="ecore:EDataType

http://www.eclipse.org/emf/2002/Ecore#//EEList">

<eTypeArguments eClassifier="#//Pasutijums"/>

</eGenericType>

</eOperations>

<eOperations name="iegutPasutijumusAkceptam">

<eGenericType eClassifier="ecore:EDataType

http://www.eclipse.org/emf/2002/Ecore#//EEList">

<eTypeArguments eClassifier="#//Pasutijums"/>

</eGenericType>

</eOperations>

<eStructuralFeatures xsi:type="ecore:EReference" name="darbinieki"

upperBound="-1" eType="#//Darbinieks" containment="true"

eOpposite="#//Darbinieks/vaditajs"/>

</eClassifiers>

<eClassifiers xsi:type="ecore:EClass" name="PrecuKatalogs">

<eOperations name="atlasit">

<eGenericType eClassifier="ecore:EDataType

http://www.eclipse.org/emf/2002/Ecore#//EEList">

<eTypeArguments eClassifier="#//Prece"/>

</eGenericType>

<eParameters name="kategorija" eType="#//Kategorija"/>

</eOperations>

<eOperations name="pievienotPreci" eType="ecore:EDataType

http://www.eclipse.org/emf/2002/Ecore#//EBoolean">

<eParameters name="prece" eType="#//Prece"/>

</eOperations>

<eOperations name="dzestPreci" eType="ecore:EDataType

http://www.eclipse.org/emf/2002/Ecore#//EBoolean">

<eParameters name="prece" eType="#//Prece"/>

</eOperations>

<eStructuralFeatures xsi:type="ecore:EReference" name="preces" upperBound="-1"

eType="#//Prece"/>

</eClassifiers>

<eClassifiers xsi:type="ecore:EClass" name="Prece">

<eOperations name="atjaunot"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="nosaukums"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EString"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="kategorija"

eType="#//Kategorija"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="vienibasCena"

eType="ecore:EDataType

http://www.eclipse.org/emf/2002/Ecore#//EBigDecimal"/>

<eStructuralFeatures xsi:type="ecore:EReference" name="precesVieniba"

eType="#//PrecesVieniba" eOpposite="#//PrecesVieniba/prece"/>

</eClassifiers>

<eClassifiers xsi:type="ecore:EClass" name="PrecesVieniba">

<eOperations name="sarekinatKopejoCenu" eType="ecore:EDataType

http://www.eclipse.org/emf/2002/Ecore#//EBigDecimal"/>

<eOperations name="noteiktCenuBezPVN" eType="ecore:EDataType

http://www.eclipse.org/emf/2002/Ecore#//EBigDecimal"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="daudzums"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EInt"/>

<eStructuralFeatures xsi:type="ecore:EReference" name="pasutijums"

lowerBound="1" eType="#//Pasutijums"

eOpposite="#//Pasutijums/precesVieniba"/>

<eStructuralFeatures xsi:type="ecore:EReference" name="prece" lowerBound="1"

eType="#//Prece" containment="true" eOpposite="#//Prece/precesVieniba"/>

</eClassifiers>

<eClassifiers xsi:type="ecore:EEnum" name="Kategorija">

<eLiterals name="Aksesuari" value="1" literal="Aksesuari"/>

<eLiterals name="BirojaPreces" value="2"/>

<eLiterals name="GaldaPiederumi" value="3"/>

<eLiterals name="Rakstampiederumi" value="4"/>

</eClassifiers>

<eClassifiers xsi:type="ecore:EClass" name="Pasutijums">

<eOperations name="akceptet"/>

<eOperations name="pasutit"/>

<eOperations name="sanemt"/>

<eOperations name="atcelt"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="numurs"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EInt"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="pasutisanasDatums"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EDate"/>

<eStructuralFeatures xsi:type="ecore:EReference" name="darbinieks"

lowerBound="1"eType="#//Darbinieks" eOpposite="#//Darbinieks/pasutijumi"/>

<eStructuralFeatures xsi:type="ecore:EReference" name="precesVieniba"

lowerBound="1" upperBound="-1" eType="#//PrecesVieniba" containment="true"

eOpposite="#//PrecesVieniba/pasutijums"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="sanemsanasDatums"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EDate"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="akceptesanasDatums"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EDate"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="status"

eType="#//PasutijumaStatus"/>

</eClassifiers>

<eClassifiers xsi:type="ecore:EEnum" name="Nodala">

<eLiterals name="Administrativa" value="1"/>

<eLiterals name="Tirzniecibas" value="2"/>

<eLiterals name="Razosanas" value="3"/>

<eLiterals name="Parvaldibas" value="4"/>

</eClassifiers>

<eClassifiers xsi:type="ecore:EEnum" name="PasutijumaStatus">

<eLiterals name="Pasutits" value="1"/>

<eLiterals name="Akceptets" value="2"/>

<eLiterals name="Sanemts" value="3"/>

<eLiterals name="Atcelts" value="4"/>

</eClassifiers>

</ecore:EPackage>

# praktiskais uzdevums



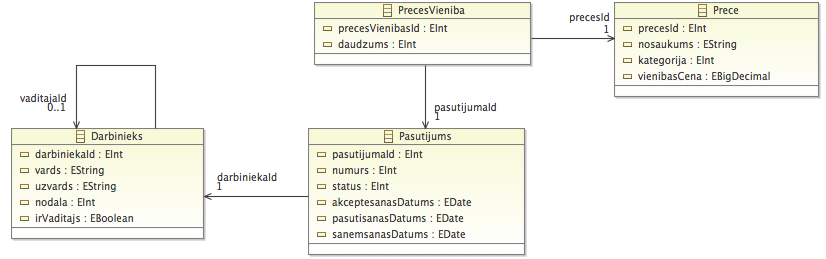
## Uzdevuma nostādne

1. Izlabot pirmo .ecore modeli pēc komentāriem.
2. Definēt datu bāzes shēmu (tabulas, primārās atslēgas un foreign keys), kura atbilst Jūsu diagrammai no 3. pr. darba. Aprakstīt to jaunā .ecore modelī.
3. Dabiskajā valodā, izmantojot elementu nosaukumus un tipus no Jūsu diviem .ecore modeļiem, definēt kartēšanas likumus no pirmā modeļa uz otro modeli.

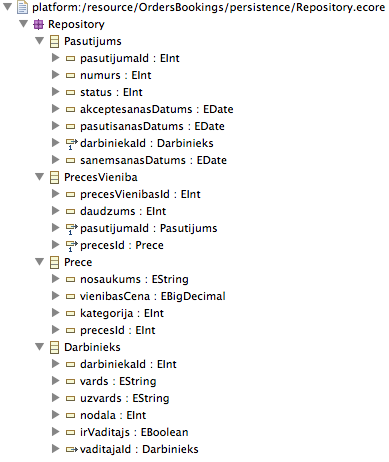
## Risinājums

Veicot uzdevuma izpildi tika izveidota 4.1 attēlā attēlota sistēmas datu bāzes shēmas diagramma. Uzdevumā ietvaros ir pārdefinētas atsauces starp klasēm, pārskatījuma tipiem ir definēts vesela skaitļa tips un vairs netiek izmantota preču kataloga klase, jo nesatur nepieciešamos atribūtus glabāšanai. Shēmas ietvaros ir paredzēts glabāt nodaļas vadītāju darbinieku tabulā, kurā ir izveidots speciāls lauks darbinieka tipa identificēšanai.

Datu bāzes shēmas ECore modelis ir attēlots 4.2 attēlā.



* 1. att. Biroja preču pasūtīšanas sistēmas datu bāzes shēmas diagramma.



* 1. att. Sistēmas datu bāzes shēmas *ECore* modelis.

Datu bāzes shēmas modeļa XML formāts izskatās sekojoši:

<?xml version="1.0" encoding="UTF-8"?>

<ecore:EPackage xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore" name="Repository">

<eClassifiers xsi:type="ecore:EClass" name="Pasutijums">

<eStructuralFeatures xsi:type="ecore:EAttribute" name="pasutijumaId"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EInt"

iD="true"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="numurs"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EInt"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="status"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EInt"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="akceptesanasDatums"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EDate"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="pasutisanasDatums"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EDate"/>

<eStructuralFeatures xsi:type="ecore:EReference" name="darbiniekaId"

lowerBound="1"

eType="#//Darbinieks" eKeys="#//Darbinieks/darbiniekaId"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="sanemsanasDatums"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EDate"/>

</eClassifiers>

<eClassifiers xsi:type="ecore:EClass" name="PrecesVieniba">

<eStructuralFeatures xsi:type="ecore:EAttribute" name="precesVienibasId"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EInt"

iD="true"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="daudzums"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EInt"/>

<eStructuralFeatures xsi:type="ecore:EReference" name="pasutijumaId"

lowerBound="1"

eType="#//Pasutijums" eKeys="#//Pasutijums/pasutijumaId"/>

<eStructuralFeatures xsi:type="ecore:EReference" name="precesId" lowerBound="1"

eType="#//Prece" eKeys="#//Prece/precesId"/>

</eClassifiers>

<eClassifiers xsi:type="ecore:EClass" name="Prece">

<eStructuralFeatures xsi:type="ecore:EAttribute" name="nosaukums"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EString"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="vienibasCena"

eType="ecore:EDataType

http://www.eclipse.org/emf/2002/Ecore#//EBigDecimal"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="kategorija"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EInt"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="precesId"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EInt"

iD="true"/>

</eClassifiers>

<eClassifiers xsi:type="ecore:EClass" name="Darbinieks">

<eStructuralFeatures xsi:type="ecore:EAttribute" name="darbiniekaId"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EInt"

iD="true"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="vards"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EString"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="uzvards"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EString"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="nodala"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EInt"/>

<eStructuralFeatures xsi:type="ecore:EAttribute" name="irVaditajs"

eType="ecore:EDataType http://www.eclipse.org/emf/2002/Ecore#//EBoolean"

defaultValueLiteral=""/>

<eStructuralFeatures xsi:type="ecore:EReference" name="vaditajaId"

eType="#//Darbinieks" eKeys="#//Darbinieks/darbiniekaId"/>

</eClassifiers>

</ecore:EPackage>

Uzdevuma ietvaros tika definēti sekojošie kartēšanas likumi sistēmas klāšu modeļa pārveidošanai datu bāzes shēmas modulī:

Jā klasei *Darbinieks* nav atribūta *darbiniekaId*, tad izveidot atribūtu *darbiniekaId* ar datu tipu *EInteger* un definēt šo atribūtu kā primāras atslēgas lauku.

IF

{S: ?x īpašība EType ir „EClass”} and

{S: ?x īpašība Name ir „Darbinieks”} and

{S: neeksistē ?x īpašība EAttribute ar Name = „darbiniekaId” }

THEN

{T: izveidot jaunu atribūtu ?y ar Name=”darbiniekaId”, EType = „EInteger” un ID=”true” } and

{T: add ?x īpašību ?y}

Jā klasei *Pasutijums* nav atribūta *pasutijumaId*, tad izveidot atribūtu *pasutijumaId* ar datu tipu *EInteger* un definēt šo atribūtu kā primāras atslēgas lauku.

IF

{S: ?x īpašība EType ir „EClass”} and

{S: ?x īpašība Name ir „Pasutijums”} and

{S: neeksistē ?x īpašība EAttribute ar Name = „pasutijumaId” }

THEN

{T: izveidot jaunu atribūtu ?y ar Name=”pasutijumaId”, EType = „EInteger” un ID=”true”} and

{T: add ?x īpašību ?y}

Jā klasei *PrecesVieniba* nav atribūta *precesVienibasId*, tad izveidot atribūtu *precesVienibasId* ar datu tipu *EInteger* un definēt šo atribūtu kā primāras atslēgas lauku.

IF

{S: ?x īpašība EType ir „EClass”} and

{S: ?x īpašība Name ir „PrecesVieniba”} and

{S: neeksistē ?x īpašība EAttribute ar Name = „precesVienibasId” }

THEN

{T: izveidot jaunu atribūtu ?y ar Name=”precesVienibasId”, EType = „EInteger” un ID=”true” } and

{T: add ?x īpašību ?y}

Jā klasei *Prece* nav atribūta *precesId*, tad izveidot atribūtu *precesId* ar *EInteger* datu tipu un definēt šo atribūtu kā primāras atslēgas lauku.

IF

{S: ?x īpašība EType ir „EClass”} and

{S: ?x īpašība Name ir „Prece”} and

{S: neeksistē ?x īpašība EAttribute ar Name = „precesId” }

THEN

{T: izveidot jaunu atribūtu ?y ar Name=”precesId”, EType = „EInteger” un ID=”true” } and

{T: add ?x īpašību ?y}

Jā klasei *Darbinieks* nav atribūta *irVaditajs*, tad izveidot atribūtu *irVaditajs* ar datu tipu *EBoolean*.

IF

{S: ?x īpašība EType ir „EClass”} and

{S: ?x īpašība Name ir „Darbinieks”} and

{S: neeksistē ?x īpašība EAttribute ar Name = „irVaditajs” }

THEN

{T: izveidot jaunu atribūtu ?y ar Name=”irVaditajs” un EType = „EBoolean” } and

{T: add ?x īpašību ?y}

Jā klase ir *Darbinieks* un tā satur atribūtu *nodala,* kuras tips nav *EInt*; vai klase ir *Prece* un tā satur atribūtu *kategorija*, kuras tips nav *EInt*; vai arī klase ir *Pasutijums* un tā satur atribūtu *status*, kuras tips nav *EInt*, tad piešķirt atribūtiem *EInt* vērtību.

IF

{S: ?x īpašība EType ir „EClass”} and

{

{{S: ?x īpašība Name ir „Prece”} and

{S: eksistē ?x īpašība EAttribute ar Name = „kategorija” and īpašības EType vērtība nav „EInt” }} or

{{S: ?x īpašība Name ir „Darbinieks”} and

{S: eksistē ?x īpašība EAttribute ar Name = „nodala” and īpašības EType vērtība nav „EInt” }}

or

{{S: ?x īpašība Name ir „Pasutijums”} and

{S: eksistē ?x īpašība EAttribute ar Name = „status” and īpašības EType vērtība nav „EInt” }}

}

THEN

{T: set ?x īpašības EAttribute īpašību EType = „EInt” }

Jā klase ir *PrecuKatalogs* vai *NodalasVaditajs*, tad dzēšam to.

IF

{S: ?x īpašība EType ir „EClass”} and

{{S: ?x īpašība Name ir „PrecuKatalogs”} or

{S: ?x īpašība Name ir „NodalasVaditajs”}}

THEN

{T: delete ?x}

Jā eksistē pārskaitījuma datu struktūras (*enumeration*), tad dzēšam tās.

IF

{S: eksistē viena vai vārākas ?x īpašības ar EType ir „EEnum”}

THEN

{T: delete visas ?x īpašības}

Jā klases *Darbinieks*, *Pasutijums*, *Prece* un *PrecuVieniba* satur metodes, tad dzēšam tās.

IF

{S: ?x īpašība EType ir „EClass”} and

{

{S: ?x īpašība Name ir ”Darbinieks”} or

{S: ?x īpašība Name ir ”Pasutijums”} or

{S: ?x īpašība Name ir ”Prece”} or

{S: ?x īpašība Name ir ”PrecuVieniba”}

} and

{S: eksistē ?x īpašība EOperation ?o}

THEN

{T: delete ?x īpašības EOperation ?o}

Jā klases *Darbinieks*, *Pasutijums*, *Prece* un *PrecuVieniba* satur asociācijas, tad dzēst tās.

IF

{S: ?x īpašība EType ir „EClass”} and

{

{S: ?x īpašība Name ir ”Darbinieks”} or

{S: ?x īpašība Name ir ”Pasutijums”} or

{S: ?x īpašība Name ir ”Prece”} or

{S: ?x īpašība Name ir ”PrecuVieniba”}

} and

{S: eksistē ?x īpašība EReference ?r}

THEN

{T: delete ?x īpašības EReference ?r}

Definējam vadītāja attiecības saiti klases *Darbinieks* ietvaros.

IF

{S: ?y īpašība EType ir „EClass”} and

{S: ?y īpašība Name ir „Darbinieks”} and

{S: eksistē ?y īpašības EType atribūts a? ar Name vienādu ar „darbiniekaId”}

THEN

{T: izveidot jaunu atribūtu ?r ar EType=”Darbinieks” un Name=”vaditajaId”} and

{T: set ?r īpašībai EKeys vērtību a?} and

{T: set ?r īpašībai LowerBound vērtību „0” un īpašībai UpperBound vērtību „1”}

{T: add ?y īpašību ?r}

Definējam vadītāja attiecības saiti starp klasēm *Darbinieks* un *Pasutijums*.

IF

{S: ?x īpašība EType ir „EClass”} and

{S: ?x īpašība Name ir „Pasutijums”} and

{S: ?y īpašība EType ir „EClass”} and

{S: ?y īpašība Name ir „Darbinieks”} and

{S: eksistē ?y īpašības EType atribūts a? ar Name vienādu ar „darbiniekaId”}

THEN

{T: izveidot jaunu atribūtu ?r ar EType=”Darbinieks” un Name=”darbiniekaId”} and

{T: set ?r īpašībai EKeys vērtību a?} and

{T: set ?r īpašībai LowerBound vērtību „1” un īpašībai UpperBound vērtību „1”}

{T: add ?x īpašību ?r}

Definējam attiecības saiti starp klasēm *Pasutijums* un *PrecesVieniba*.

IF

{S: ?x īpašība EType ir „EClass”} and

{S: ?x īpašība Name ir „PrecesVieniba”} and

{S: ?y īpašība EType ir „EClass”} and

{S: ?y īpašība Name ir „Pasutijums”} and

{S: eksistē ?y īpašības EType atribūts a? ar Name vienādu ar „pasutijumaId”}

THEN

{T: izveidot jaunu atribūtu ?r ar EType=”Pasutijums” un Name=”pasutijumaId”} and

{T: set ?r īpašībai EKeys vērtību a?} and

{T: set ?r īpašībai LowerBound vērtību „1” un īpašībai UpperBound vērtību „1”}

{T: add ?x īpašību ?r}

Definējam attiecības saiti starp klasēm *Prece* un *PrecesVieniba*.

IF

{S: ?x īpašība EType ir „EClass”} and

{S: ?x īpašība Name ir „PrecesVieniba”} and

{S: ?y īpašība EType ir „EClass”} and

{S: ?y īpašība Name ir „Prece”} and

{S: eksistē ?y īpašības EType atribūts a? ar Name vienādu ar „precesId”}

THEN

{T: izveidot jaunu atribūtu ?r ar EType=”Prece” un Name=”precesId”} and

{T: set ?r īpašībai EKeys vērtību a?} and

{T: set ?r īpašībai LowerBound vērtību „1” un īpašībai UpperBound vērtību „1”}

{T: add ?x īpašību ?r}

# praktiskais uzdevums

## Uzdevuma nostādne

1. Izveidot XML failu, kas satur 3. praktiskajā darbā izstrādātā un 4. praktiskajā darbā izlabotā modeļa instanci ar definētajiem klašu atribūtiem piešķirtām vērtībām.
2. Izstrādāt ATL valodā transformāciju. Transformācijas ieejas modelis ir augstākminētais XML fails ar datiem, tā metamodelis ir augstākminētais .ecore modelis. Transformācijas izejas modelim jābūt citam XML failam. Tā metamodelis ir 4. praktiskajā darbā izveidots .ecore modelis ar datubāzes shēmu. Transformācijas rezultātā jaunizveidotam XML failam jāsatur datubāzes shēmai atbilstoši pārveidotie dati.

## Risinājums

Balstoties uz 3. praktiskajā darbā izstrādāta modeļa instanci ir izstrādāts sekojošs XML fails:

<?xml version="1.0" encoding="UTF8"?>

<xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns="Bookings">

<NodalasVaditajs vards="Homer" uzvards="Simpson" nodala="Parvaldibas">

<darbinieki vards="Bart" uzvards="Simpson" nodala="Razosanas">

<pasutijumi numurs="7309909" status="Akceptets"

akceptesanasDatums="2015-05-27 15:33"

pasutisanasDatums="2015-05-27 16:00">

<precesVieniba daudzums="1">

<prece nosaukums="Cool kattle" kategorija="BirojaPreces"

vienibasCena="48"/>

</precesVieniba>

</pasutijumi>

<pasutijumi numurs="1770313" status="Atcelts"

pasutisanasDatums="2015-05-29 12:03">

<precesVieniba daudzums="1">

<prece nosaukums="Mobile charger" kategorija="Aksesuari"

vienibasCena="3.99"/>

</precesVieniba>

</pasutijumi>

</darbinieki>

<darbinieki vards="Lisa" uzvards="Simpson" nodala="Tirzniecibas">

<pasutijumi numurs="6740736" status="Pasutits"

pasutisanasDatums="2015-06-1 10:00">

<precesVieniba daudzums="3">

<prece nosaukums="Pen" kategorija="Rakstampiederumi"

vienibasCena="0.78"/>

</precesVieniba>

<precesVieniba daudzums="5">

<prece nosaukums="Pencil" kategorija="Rakstampiederumi"

vienibasCena="0.55"/>

</precesVieniba>

</pasutijumi>

</darbinieki>

</NodalasVaditajs>

<NodalasVaditajs vards="Jack" uzvards="Sparrow" nodala="Parvaldibas">

<darbinieki vards="Elizabeth" uzvards="Swann" nodala="Administrativa">

<pasutijumi numurs="0814640" status="Sanemts"

pasutisanasDatums="2015-05-01 17:46"

akceptesanasDatums="2015-05-02 11:15"

sanemsanasDatums="2015-05-09 10:00">

<precesVieniba daudzums="1">

<prece nosaukums="Lamp" kategorija="GaldaPiederumi"

vienibasCena="10.09"/>

</precesVieniba>

</pasutijumi>

</darbinieki>

</NodalasVaditajs>

<NodalasVaditajs vards="Santa" uzvards="Claus" nodala="Parvaldibas">

<pasutijumi numurs="4156674" status="Akceptets"

pasutisanasDatums="2015-05-31 13:46"

akceptesanasDatums="2015-05-31 13:48">

<precesVieniba daudzums="1">

<prece nosaukums="Santa suite" kategorija="Aksesuari"

vienibasCena="20.99"/>

</precesVieniba>

</pasutijumi>

</NodalasVaditajs>

</xmi:XMI>

Failā tiek definēti dati nodaļas vadītājiem, to pasūtījumiem un darbiniekiem, kuriem ir raksturīgs savs pasūtījumu kopums. Katrām pasūtījumam atbilst noteikta vienība, kurā ir definēta prece un tās kopējais daudzums.

Lai pārveidotu ienākoša faila struktūru un tā datus atbilstoši 4. praktiskajā darbā izveidotām metamodelim, ir izstrādāts 5.1 attēlā attēlots ATL valodas pirmteksts.

|  |  |
| --- | --- |
| 01 | -- @path Bookings=/OrdersBookings/model/Bookings.ecore |
| 02 | -- @path Repository=/OrdersBookings/persistence/Repository.ecore |
| 03 |  |
| 04 | **module** Bookings2Repository; |
| 05 | **create** OUT : Repository **from** IN : Bookings; |
| 06 |  |
| 07 | **helper** **def**: lastUnitIdentity : Integer = 0; |
| 08 | **helper** **def**: lastOrderIdentity : Integer = 0; |
| 09 | **helper** **def**: lastProductIdentity : Integer = 0; |
| 10 | **helper** **def**: lastEmployeeIdentity : Integer = 0; |
| 11 |  |
| 12 | **helper** **context** Bookings!Darbinieks **def**: isManager(): Boolean = |
| 13 | **if** **self**.oclIsTypeOf(Bookings!NodalasVaditajs) **then** |
| 14 | true |
| 15 | **else** |
| 16 | false |
| 17 | **endif**; |
| 18 |  |
| 19 | **helper** **def**: getDepartmentValue(department: Bookings!Nodala): Integer = |
| 20 | **let** departments: Sequence(Bookings!Nodala) = |
| 21 | Sequence { #Administrativa, #Tirzniecibas, #Razosanas, #Parvaldibas } |
| 22 | **in** departments.indexOf(department); |
| 23 |  |
| 24 | **helper** **def**: getCategoryValue(category: Bookings!Kategorija): Integer = |
| 25 | **let** categories: Sequence(Bookings!Kategorija) = |
| 26 | Sequence { #Aksesuari, #BirojaPreces, #GaldaPiederumi, #Rakstampiederumi } |
| 27 | **in** categories.indexOf(category); |
| 28 |  |
| 29 | **helper** **def**: getStatusValue(status: Bookings!PasutijumaStatus): Integer = |
| 30 | **let** statuses: Sequence(Bookings!PasutijumaStatus) = |
| 31 | Sequence { #Jauns, #Akceptets, #Pasutits, #Sanemts, #Atcelts } |
| 32 | **in** statuses.indexOf(status); |
| 33 |  |
| 34 | **rule** LinkEmployee { |
| 35 | **from** |
| 36 | s: Bookings!Darbinieks |
| 37 | **to** |
| 38 | t: Repository!Darbinieks ( |
| 39 | vards <- s.vards, |
| 40 | uzvards <- s.uzvards, |
| 41 | vaditajaId <- s.vaditajs, |
| 42 | irVaditajs <- s.isManager(), |
| 43 | nodala <- **thisModule**.getDepartmentValue(s.nodala) |
| 44 | ) |
| 45 | **do** { |
| 46 | **thisModule**.lastEmployeeIdentity <- **thisModule**.lastEmployeeIdentity + 1; |
| 47 | t.darbiniekaId <- **thisModule**.lastEmployeeIdentity; |
| 48 | } |
| 49 | } |
| 50 |  |
| 51 | **rule** LinkOrder { |
| 52 | **from** |
| 53 | s: Bookings!Pasutijums |
| 54 | **to** |
| 55 | t: Repository!Pasutijums ( |
| 56 | numurs <- s.numurs, |
| 57 | darbiniekaId <- s.darbinieks, |
| 58 | sanemsanasDatums <- s.sanemsanasDatums, |
| 59 | pasutisanasDatums <- s.pasutisanasDatums, |
| 60 | akceptesanasDatums <- s.akceptesanasDatums, |
| 61 | status <- **thisModule**.getStatusValue(s.status) |
| 62 | ) |
| 63 | **do** { |
| 64 | **thisModule**.lastOrderIdentity <- **thisModule**.lastOrderIdentity + 1; |
| 65 | t.pasutijumaId <- **thisModule**.lastOrderIdentity; |
| 66 | } |
| 67 | } |
| 68 |  |
| 69 | **rule** LinkProduct { |
| 70 | **from** |
| 71 | s: Bookings!Prece |
| 72 | **to** |
| 73 | t: Repository!Prece ( |
| 74 | nosaukums <- s.nosaukums, |
| 75 | vienibasCena <- s.vienibasCena, |
| 76 | kategorija <- **thisModule**.getCategoryValue(s.kategorija) |
| 77 | ) |
| 78 | **do** { |
| 79 | **thisModule**.lastProductIdentity <- **thisModule**.lastProductIdentity + 1; |
| 80 | t.precesId <- **thisModule**.lastProductIdentity; |
| 81 | } |
| 82 | } |
| 83 |  |
| 84 | **rule** LinkUnit { |
| 85 | **from** |
| 86 | s: Bookings!PrecesVieniba |
| 87 | **to** |
| 88 | t: Repository!PrecesVieniba ( |
| 89 | daudzums <- s.daudzums, |
| 90 | precesId <- s.prece, |
| 91 | pasutijumaId <- s.pasutijums |
| 92 | ) |
| 93 | **do** { |
| 94 | **thisModule**.lastUnitIdentity <- **thisModule**.lastUnitIdentity + 1; |
| 95 | t.precesVienibasId <- **thisModule**.lastUnitIdentity; |
| 96 | } |
| 97 | } |

* 1. att. ATL valodas pirmteksts.

Transformācijas izpildes rezultātā tiek veidots sekojošs XML fails:

<?xml version="1.0" encoding="ISO-8859-1"?>

<xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns="Repository">

<Darbinieks darbiniekaId="1" vards="Homer" uzvards="Simpson" nodala="4"

irVaditajs="true"/>

<Darbinieks darbiniekaId="2" vards="Bart" uzvards="Simpson" nodala="3"

irVaditajs="false" vaditajaId="1"/>

<Darbinieks darbiniekaId="3" vards="Lisa" uzvards="Simpson" nodala="2"

irVaditajs="false" vaditajaId="1"/>

<Darbinieks darbiniekaId="4" vards="Jack" uzvards="Sparrow" nodala="4"

irVaditajs="true"/>

<Darbinieks darbiniekaId="5" vards="Elizabeth" uzvards="Swann" nodala="1"

irVaditajs="false" vaditajaId="4"/>

<Darbinieks darbiniekaId="6" vards="Santa" uzvards="Claus" nodala="4"

irVaditajs="true"/>

<Pasutijums pasutijumaId="1" numurs="7309909" status="2"

akceptesanasDatums="2015-05-27T00:00:00.000+0300"

pasutisanasDatums="2015-05-27T00:00:00.000+0300" darbiniekaId="2"/>

<Pasutijums pasutijumaId="2" numurs="1770313" status="5"

pasutisanasDatums="2015-05-29T00:00:00.000+0300" darbiniekaId="2"/>

<Pasutijums pasutijumaId="3" numurs="6740736" status="3"

pasutisanasDatums="2015-06-01T00:00:00.000+0300" darbiniekaId="3"/>

<Pasutijums pasutijumaId="4" numurs="814640" status="4"

akceptesanasDatums="2015-05-02T00:00:00.000+0300"

pasutisanasDatums="2015-05-01T00:00:00.000+0300" darbiniekaId="5"

sanemsanasDatums="2015-05-09T00:00:00.000+0300"/>

<Pasutijums pasutijumaId="5" numurs="4156674" status="2"

akceptesanasDatums="2015-05-31T00:00:00.000+0300"

pasutisanasDatums="2015-05-31T00:00:00.000+0300" darbiniekaId="6"/>

<Prece nosaukums="Cool kattle" vienibasCena="48" kategorija="2" precesId="1"/>

<Prece nosaukums="Mobile charger" vienibasCena="3.99" kategorija="1"

precesId="2"/>

<Prece nosaukums="Pen" vienibasCena="0.78" kategorija="4" precesId="3"/>

<Prece nosaukums="Pencil" vienibasCena="0.55" kategorija="4" precesId="4"/>

<Prece nosaukums="Lamp" vienibasCena="10.09" kategorija="3" precesId="5"/>

<Prece nosaukums="Santa suite" vienibasCena="20.99" kategorija="1" precesId="6"/>

<PrecesVieniba precesVienibasId="1" daudzums="1" pasutijumaId="1" precesId="1"/>

<PrecesVieniba precesVienibasId="2" daudzums="1" pasutijumaId="2" precesId="2"/>

<PrecesVieniba precesVienibasId="3" daudzums="3" pasutijumaId="3" precesId="3"/>

<PrecesVieniba precesVienibasId="4" daudzums="5" pasutijumaId="3" precesId="4"/>

<PrecesVieniba precesVienibasId="5" daudzums="1" pasutijumaId="4" precesId="5"/>

<PrecesVieniba precesVienibasId="6" daudzums="1" pasutijumaId="5" precesId="6"/>

</xmi:XMI>

# praktiskais uzdevums

## Uzdevuma nostādne

1. Izstrādāt transformācijas uz programmēšanas valodas (Java, C# vai C/C++) konstrukcijām. Transformējot pieņemam, ka pēc noklusēšanas visas klases ir publiskās, klašu atribūti ar primitīviem tipiem ir aizsargātie (private, protected, private protected un tml.), klašu atribūti ar citu klašu tipiem ir neaizsargātie.
2. Katrai klasei jāatrodas atsevišķajā failā (ja to paredz programmēšanas valoda). Klasēm jāizveido:
   1. konstruktori un destruktori (ja tos paredz programmēšanas valoda) vai līdzīgas metodes piekļūšanai aizsargātajiem atribūtiem,
   2. atribūti ar vērtībām, ja ir,
   3. attiecības starp klasēm:
      1. Ja starp klasēm ir mantošana, tad tas arī jāparada (jāņem vērā arī vai klase bija abstrakta).
      2. Tāpat jāpadomā par agregāciju un kompozīciju (ja programmēšanas valoda paredz kādu atšķirību mainīgo deklarācijā).

## Risinājums

Uzdevuma ietvaros tiek veikta Java programmēšanas valodas klašu un pārskaitījumu (*enum*) transformācijas, balstoties uz 3. praktiskajā darba izveidoto Ecore metamodeli. Transformāciju veikšanai ir izveidots Acceleo modeļa transformācijas valodas (MTL) fails, kurā pirmteksts ir izskatās sekojoši:

[comment encoding = UTF-8 /]

**[module** generate('http://www.eclipse.org/emf/2002/Ecore'**)]**

[comment Ecore metamodelis tiek parveidots uz Java valodas parskaitijumiem /]

**[template** **public** generateElement**(**anEnum : EEnum**)]**

**[*let*** enumUpperFirstName: String = anEnum.name.toUpperFirst(**)]**

**[*let*** ordinalVariableName: String = 'value'**]**

[comment Tiek definets parskatijuma tips /]

[comment **@main** /]

**[file (**enumUpperFirstName.concat('.java'), false, 'UTF-8'**)]**

public enum **[**enumUpperFirstName**/]** {

[comment Tiek sastadits literalu saraksts /]

**[*for* (**literal : EEnumLiteral | anEnum.eLiterals**) *separator*(**',\n'**) *after*(**';\n'**)]**

**[**literal.name.toUpper()**/]**(**[**literal.value**/]**)**[/*for*]**

[comment Tiek veidots literala vertibas tips /]

private int **[**ordinalVariableName**/]**;

**[**enumUpperFirstName**/]**(int **[**ordinalVariableName**/]**) {

this.**[**ordinalVariableName**/]** = **[**ordinalVariableName**/]**;

}

public int get**[**ordinalVariableName.toUpperFirst()**/]**() {

return **[**ordinalVariableName**/]**;

}

}

**[/file]**

**[/*let*]**

**[/*let*]**

**[/template]**

[comment Ecore metamodelis tiek parveidots uz Java valodas klasem /]

**[template** **public** generateElement**(**anClass : EClass**)]**

**[*let*** classUpperFirstName: String = anClass.name.toUpperFirst(**)]**

[comment Tiek defineta klase, ieverojot mantosanas konstrukcijas /]

[comment **@main** /]

**[file (**classUpperFirstName.concat('.java'), false, 'UTF-8'**)]**

public **[*if*** **(**anClass.abstract**)]**abstract **[/*if*]**class **[**classUpperFirstName**/]**

**[**anClass.collectSuperTypes()**/]** {

[comment Tiek veidots atributu saraksts /]

**[*if*** **(**anClass.eAttributes->size() > 0**)]**

**[*for* (**attribute : EAttribute | anClass.eAttributes**)]**

private **[**attribute.getObjectType()**/]** **[**attribute.name.toLowerFirst()**/]**;

**[/*for*]**

**[/*if*]**

[comment Tiek veidots atsaucu saraksts /]

**[*for* (**reference : EReference | anClass.eReferences**) *after*(**'\n'**)]**

private **[**reference.getReferenceTypes()**/]** **[**reference.name.toLowerFirst()**/]**;

**[/*for*]**

[comment Tiek veidots nokluseja konstruktors bez parametriem /]

public **[**classUpperFirstName**/]**() {}

[comment Tiek veidots konstruktors super klases atributu inicializacijai /]

**[*if*** **(**anClass.eAllSuperTypes->size() > 0**)]**

public **[**classUpperFirstName**/]**(**[**anClass.collectAllParams()**/]**) {

super(**[**anClass.collectAllParamNames()**/]**);

}

**[/*if*]**

[comment Tiek veidots konstruktors klases atributu inicializacijai /]

**[*if*** **(**anClass.eAttributes->size() > 0**)]**

public **[**classUpperFirstName**/]**(**[**anClass.collectParams()**/]**) {

**[*for* (**attribute : EAttribute | anClass.eAttributes**)]**

**[*let*** lowerFirstName: String = attribute.name.toLowerFirst(**)]**

this.**[**lowerFirstName**/]** = **[**lowerFirstName**/]**;

**[/*let*]**

**[/*for*]**

}

[comment Tiek veidotas atributu piekluves metodes /]

**[*for* (**attribute : EAttribute | anClass.eAttributes**) *separator*(**'\n'**)]**

**[*let*** attributeType: String = attribute.getObjectType(**)]**

**[*let*** lowerFirstName: String = attribute.name.toLowerFirst(**)]**

**[*let*** upperFirstName: String = attribute.name.toUpperFirst(**)]**

public void set**[**upperFirstName**/]**(**[**attributeType**/]** **[**lowerFirstName**/]**) {

this.**[**lowerFirstName**/]** = **[**lowerFirstName**/]**;

}

public **[**attributeType**/]** get**[**upperFirstName**/]**() {

return **[**lowerFirstName**/]**;

}

**[/*let*]**

**[/*let*]**

**[/*let*]**

**[/*for*]**

**[/*if*]**

[comment Tiek veidotas atsaucu tipu piekluves metodes /]

**[*if*** **(**anClass.eReferences->size() > 0**)]**

**[*for* (**reference : EReference | anClass.eReferences**) *after*(**'\n'**)]**

**[*let*** referenceTypes: String = reference.getReferenceTypes(**)]**

**[*let*** lowerFirstName: String = reference.name.toLowerFirst(**)]**

**[*let*** upperFirstName: String = reference.name.toUpperFirst(**)]**

public void set**[**upperFirstName**/]**(**[**referenceTypes**/]** **[**lowerFirstName**/]**) {

this.**[**lowerFirstName**/]** = **[**lowerFirstName**/]**;

}

public **[**referenceTypes**/]** get**[**upperFirstName**/]**() {

return **[**lowerFirstName**/]**;

}

**[/*let*]**

**[/*let*]**

**[/*let*]**

**[/*for*]**

**[/*if*]**

[comment Tiek veidotas metodes /]

**[*for* (**operation : EOperation | anClass.eAllOperations**) *separator*(**'\n'**) *after*(**''**)]**

**[*if*** **(**operation.eType.oclIsUndefined()**)]**

public void **[**operation.name.toLowerFirst()**/]**() {

}

**[*else*]**

public **[**operation.eType.instanceTypeName**/][**operation.checkForGenericType()**/]** **[**operation.name.toLowerFirst()**/]**(**[**operation.collectParams()**/]**) {

throw new UnsupportedOperationException();

}

**[/*if*]**

**[/*for*]**

}

**[/file]**

**[/*let*]**

**[/template]**

[comment Apkopo super klases, izpildes piem. - 'extends Darbinieks' /]

**[template** **private** collectSuperTypes**(**clazz: EClass**)]**

**[**clazz.eAllSuperTypes->collect(' extends ' + name.toUpperFirst())**/]**

**[/template]**

[comment Iegust attributa datu tipu, izpildes piem. - java.math.BigInteger /]

**[template** **private** getObjectType**(**attribute: EAttribute**)]**

**[*if*** **(**attribute.eAttributeType.instanceTypeName.oclIsUndefined()**)][**eAttributeType.name**/][*else*][**eAttributeType.instanceTypeName**/][/*if*]**

**[/template]**

[comment Apkopo atributus signaturas veida, izpildes piem. - String vards, String uzvards /]

**[template** **private** collectParams**(**clazz: EClass**)]**

**[**clazz.eAttributes->collect(getObjectType() + ' ' + name.toLowerFirst())->sep(', ')

**/]**

**[/template]**

[comment Megina iegut visparinatu tipu, izpildes piem. - <T> /]

**[template** **private** checkForGenericType**(**operation: EOperation**)]**

**[*if*** **(not** operation.eGenericType.eTypeArguments->isEmpty()**)]**<

**[**operation.eGenericType.eTypeArguments.eClassifier.name**/]**>**[/*if*]**

**[/template]**

[comment Apkopo metodes parametrus signaturas veida, izpildes piem. - Prece prece /]

**[template** **private** collectParams**(**operation: EOperation**)]**

**[**operation.eParameters->collect(eType.name + ' ' + name.toLowerFirst())->sep(', ')

**/]**

**[/template]**

[comment Apkopo gan super klases, gan pasas klases atributus signaturas veida, izpildes piem. - Nodala nodala /]

**[template** **private** collectAllParams**(**clazz: EClass**)]**

**[**clazz.eAllAttributes->collect(getObjectType() + ' ' + name)->sep(', ')**/]**

**[/template]**

[comment Apkopo gan super klases, gan pasas klases atributu nosaukums, izpildes piem. - vards, uzvards /]

**[template** **private** collectAllParamNames**(**clazz: EClass**)]**

**[**clazz.eAllAttributes->collect(name.toLowerFirst())->sep(', ')**/]**

**[/template]**

[comment Iegust atsauces objekta tipu, izpildes piem., Set<T> /]

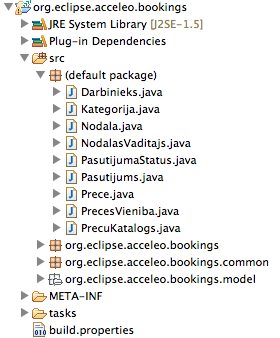
**[template** **private** getReferenceTypes**(**reference: EReference**)]**

**[*if*** **(**reference.many**)]**java.util.Set<**[**reference.eType.name**/]**>**[*else*]**

**[**reference.eType.name**/][/*if*]**

**[/template]**

Izpildes rezultātā tiek ģenerēti vairāki Java valodas pirmteksta faili, kuri attēloti 6.1 attēlā. Klases pirmteksta piemērs ir parādīts 6.2 attēlā un pārskaitījuma piemērs ir parādīts 6.3 attēlā.



* 1. att. Iegūtais rezultāts.

|  |  |
| --- | --- |
| 01 | **public** **class** NodalasVaditajs **extends** Darbinieks { |
| 02 |  |
| 03 | **private** java.util.Set<Darbinieks> darbinieki; |
| 04 |  |
| 05 | **public** NodalasVaditajs() {} |
| 06 |  |
| 07 | **public** NodalasVaditajs(java.lang.String vards, java.lang.String uzvards, |
| 08 | Nodala nodala) { |
| 09 | **super**(vards, uzvards, nodala); |
| 10 | } |
| 11 |  |
| 12 | **public** **void** setDarbinieki(java.util.Set<Darbinieks> darbinieki) { |
| 13 | **this**.darbinieki = darbinieki; |
| 14 | } |
| 15 |  |
| 16 | **public** java.util.Set<Darbinieks> getDarbinieki() { |
| 17 | **return** darbinieki; |
| 18 | } |
| 19 |  |
| 20 | **public** **void** parceltUzNodalu() { |
| 21 |  |
| 22 | } |
| 23 |  |
| 24 | **public** **void** atlaist() { |
| 25 |  |
| 26 | } |
| 27 |  |
| 28 | **public** org.eclipse.emf.common.util.EList<Pasutijums> iegutAkceptetosPasutijumus(){ |
| 29 | **throw** **new** UnsupportedOperationException(); |
| 30 | } |
| 31 |  |
| 32 | **public** org.eclipse.emf.common.util.EList<Pasutijums> iegutPasutijumusAkceptam() { |
| 33 | **throw** **new** UnsupportedOperationException(); |
| 34 | } |
| 35 | } |

* 1. att. Iegūtas klases piemērs.

|  |  |
| --- | --- |
| 01 | **public** **enum** Nodala { |
| 02 |  |
| 03 | *ADMINISTRATIVA*(1), |
| 04 | *TIRZNIECIBAS*(2), |
| 05 | *RAZOSANAS*(3), |
| 06 | *PARVALDIBAS*(4); |
| 07 |  |
| 08 | **private** **int** value; |
| 09 |  |
| 10 | Nodala(**int** value) { |
| 11 | **this**.value = value; |
| 12 | } |
| 13 |  |
| 14 | **public** **int** getValue() { |
| 15 | **return** value; |
| 16 | } |
| 17 | } |
| 35 |  |

* 1. att. Pārskaitījuma piemērs.

# Secinājumi

Kursā ietvaros ir iegūts priekšstats par metamodelēšanu un transformācijām.   
Ir iegūtas praktiskas iemaņas Ecore modeļu izstrāde, ir veiktas transformācijas ar ATL un MOFM2T rīku un valodu palīdzību.

Veicot darbu ir rādies priekšstats par metamodeļu un transformāciju lietderību sistēmas projektēšanas posmā, ļaujot izstrādāt no platformām neatkarīgo sistēmas modeli, kurš turpmāk var tikt translēts jebkura piemērota programmēšanas valodā. Tas ļauj koncentrēties uz sistēmas detaļām abstrahējoties no platformu īpatnībām. Ir izsecināta modeļu priekšrocība sistēmas datu modelēšanas jomā, kas ļauj atveidot datu saistības starp dažādam paradigmām, piemēram, datu struktūru attēlojums relāciju un objektu modelī.

Uzskatu, ka priekšmeta apgūšana ļauj apgūt vērtīgas zināšanas, kas turpmāk var tikt pielietotas veicot sistēmas izstrādi.