





**IEGULDĪJUMS TAVĀ NĀKOTNĒ!**

Eiropas Reģionālās attīstības fonds

Prioritāte: 2.1. Zinātne un inovācijas

Pasākums: 2.1.1. Zinātne, pētniecība un attīstība

Aktivitāte: 2.1.1.1. Atbalsts zinātnei un pētniecībai

# Projekts: ”Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem”

Projekta sākuma datums: 2014.gada 1.janvāris.

Projekta beigu datums: 2015.gada 30.jūnijs.

Līguma Nr. 2013/0031/2DP/2.1.1.1.0/13/APIA/VIAA/010

ESF finansējuma saņēmējs: SIA, SWH SETS

Sadarbības partneris: Elektronikas un datorzinātņu institūts (EDI)

# Projekta aktivitātes Nr.3.8.2 "Funkciju un objektorientētu lietotāju tiesību modelis" progresa pārskats

Pārskats Nr. 30 par periodu no 2014.gada 1.jūlija līdz 2014.gada 31.decembrim.

# SATURS

# 1. Kopsavilkums

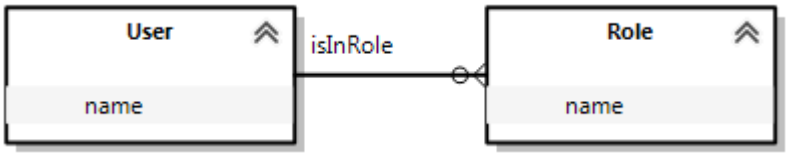
Pārskata periodā (2015-01-01 – 2015-06-30.) projekta „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" aktivitātes Nr.3.8.2 "Funkciju un objektorientētu lietotāju tiesību modelis" ietvaros veikti šādi darbi:

1. Lietotāju tiesību standart modeļu izpēte.
2. Loģiskā lietotāju tiesību modeļa izstrāde.
3. Izpildes laika modeļa izstrāde.
4. Tiesību modeļa izstrāde.
5. Tiesību tehnoloģiskā modeļa automātiskās ražošanas transformācijas pārveide.
6. Aktivitātes pētnieciskā darbība apspriesta ik nedēļas projekta semināros.

# 2. Lietotāju tiesību standartmodeļi

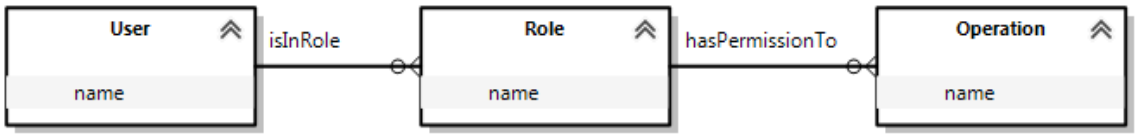
Lietojumprogrammās, kurām ir viens lietotājs nav vajadzības organizēt lietotāja tiesību kontroli – lietotājam ir pieejamas visas funkcijas. Daudzlietotāju lietojumprogrammās parasti ne visa funkcionalitāte ir pieejama visiem lietotājiem. Noteiktam lietotājam ir pieejamas tikai tās funkcijas, kuru izpildei viņam ir dotas tiesības.
Runājot par lietotāju tiesību kontroli, eksistē dažādas lietotāju pieejas tiesību tipu klasifikācijas. Piemēram National Institute of Standards and Technology ir definējis divas pieejas: uz lomām bāzētas lietotāju tiesības(Role Based Access Control - RBAC) [1] un uz atribūtiem bāzētas lietotāju tiesības (Attribute Based Access Control - ABAC) [2].
Micrsoft, savukārt, lieto terminu Claims Based Access Control, kas pēc būtības ir līdzīgs ABAC [3].
RBAC ir autorizācijas pieeja, kurā lietotāju tiesības tiek piešķirtas un pārbaudītas, izmantojot lietotāju lomas. Ja lietotājam ir loma, kura nepieciešama kādas darbības veikšanai, tad pieeja šai darbībai tiek atļauta, pretējā gadījumā lietotajam nav pieejas darbībai. CBAC un ABAC gadījumā, lēmums piešķirt vai noliegt atļauju tiek pieņemts atkarībā no kaut kādu atribūtu, kuri piešķirti lietotājam, vērtību kopas.

Nākošajā zīmējumā redzams pats vienkāršākais RBAC modelis:




**Zīmējums 1**

Modelī ir lietotājs (User) un loma (Role). Izmantojot šo modeli, tiesību pārbaude tiek stingri ierakstīta programmas kodā, kas lietotāja tiesību loģiku ļoti stingri piesaista pie biznesa loģikas. Tas bieži vien rada problēmas uzturot un mainot informatīvo sistēmu.
Sīkāk šī problēma ir aprakstīta rakstā [4].
Tādēļ bieži lietotāju tiesību modelī ievieš vēl vienu entītiju, kas apraksta lietotāja tiesības operācijai (aktivitātei). Dažādos avotos to mēdz saukt dažādi Operation, Permission, Task vai Activity. Mēs turpmāk tekstā to sauksim par operāciju (Operation):




**Zīmējums 2**

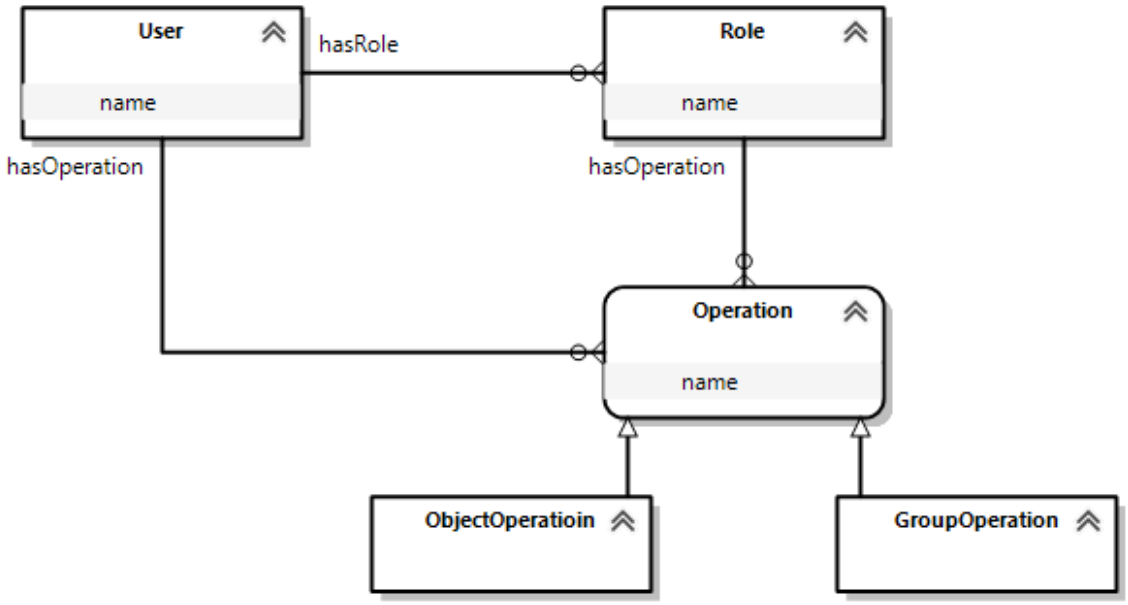
Šāda pieeja ļauj pārbaudīt lietotāja tiesības funkcionālā, ne lomu līmenī. Tipisks piemērs, kā izmantot šādu pieeju ir CRUD (Create,Read,Update,Delete) operācijas noteiktai datubāzes tabulai.

# 3. Izvēlētais lietotāju tiesību modelis

Lietotāju tiesību modelis sastāv no divām daļām: loģiskā modeļa un izpildes laika modeļa. Loģiskais modelis ir balstīts uz RBAC un apraksta lietotāju tiesību pamatjēdzienus. Izpildes laika modelis apraksta, kā tiek pārbaudītas lietotāja tiesības programmas izpildes laikā.

## 3.1. Loģiskais lietotāju tiesību modelis

Loģiskais modelis sevī ietver iepriekšējā sadaļā aprakstītos jēdzienus loma (Role) un operācija (Operation). Lai padarītu modeli universālāku, tas paredz, ka tiesības uz operāciju var piešķirt arī konkrētam lietotājam.




**Zīmējums 3**

Atbilstoši šim modelim informācija par lietotāju un viņa tiesībām tiek glabāta datubāzē. Operācijai (Operation) ir divi apakštipi – ObjectOperation un GroupOperation. ObjectOperation ir operācija, kas ir piesaistīta noteiktam objekta tipam, GroupOperation ir operācija, kas piesaistīta vairākiem objektu tipiem. Lai saprastu, kā notiek operāciju piesaiste objektu tipiem, apskatīsim izpildes laika modeli.

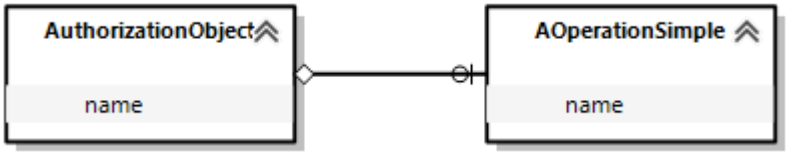
## 3.2. Izpildes laika modelis

Tālāk apskatīsim lietotāju tiesību izpildes laika modeli. Šis modelis parāda, kā izpildes laikā tiks pārbaudītas lietotāju tiesības.
Lai vieglāk saprastu izvēlēto lietotāju tiesību modeli, sakumā apskatīsim piemēru, uzzīmēsim tā modeli, pamazām papildināsim piemēru, lai iegūtu universālāku modeli.
Pieņemsim, ka mums ir DataView no [5] ar operācijām: Create, Read, Update un Delete.

Teiksim, ka šīs ir elementāras operācijas ar tipu AOperationSimple un uzskatīsim, ka lietotājam var tikt dotas tiesības veikt šīs operācijas.


**Zīmējums 4**

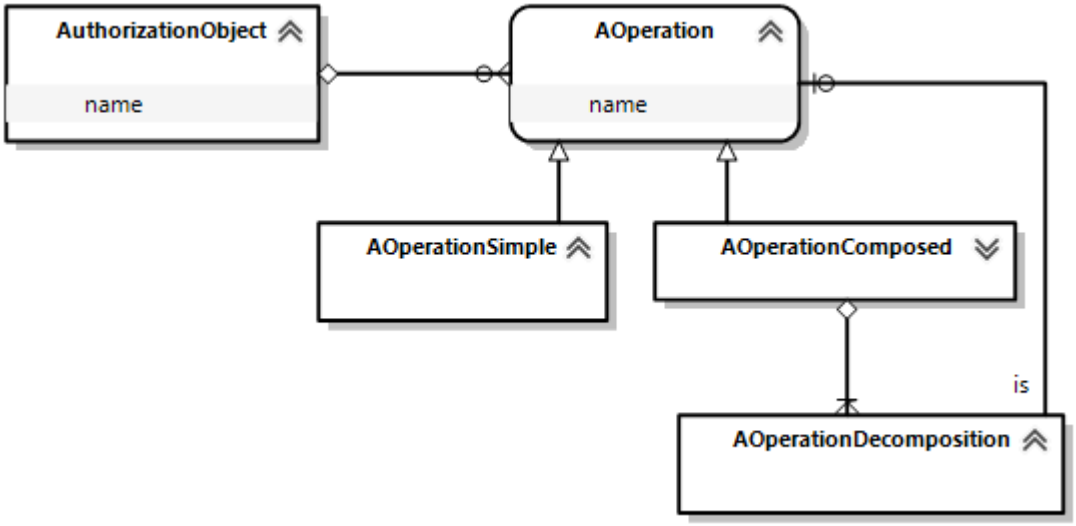
Uzskatīsim, ka DataView vietā var būt jebkurš objekts no mūsu metamodeļiem un AOperationSimple ir elementāra darbība ar šo objektu (piemēram, objekta radīšana (Create) vai arī funkcija, kuru mēs pielietojam objektam), kurai mēs gribam atļaut vai neatļaut lietotāja pieeju.




**Zīmējums 5**

Tātad **AOperationSimple** varētu būt jebkura elementāra operācija, kura tiek veikta ar **AuthorizationObject** un kuru mēs vēlamies atļaut vai neatļaut izpildīt lietotājam.

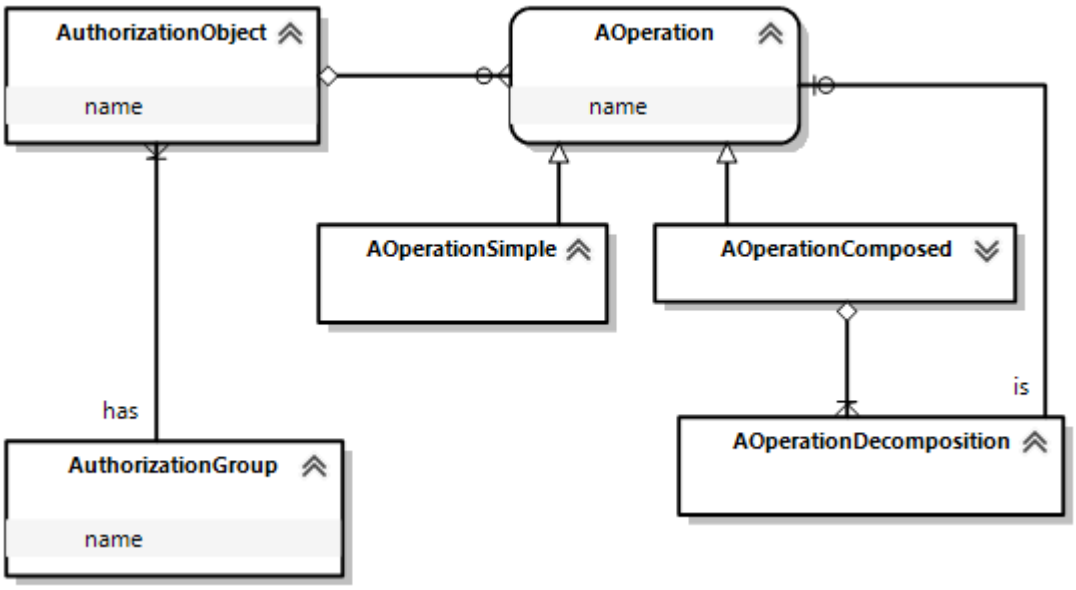
Uzskatīsim, ka mums ir sarežģītākas operācijas, kuru izpildei lietotājam ir nepieciešamas tiesības izpildīt citas operācijas un teiksim, ka šādām operācijām ir tips **AOperationComposed**.




**Zīmējums 6**

**AOperationDecomposition** redzamajā modelī kalpo kā saite, lai norādītu uz operāciju, kuras pieejas tiesības ir nepieciešamas, lai drīkstētu izpildīt AOperationComposed. Faktiski modelis attēlo operāciju koku, kura lapas ir elementāras operācijas.

Papildināsim modeli ar **AuthorizationGroup**, kas ļauj sagrupēt objektus. Piemēram, visas tabulas, kas ir klasifikatori varētu atrasties grupā "UpdateClassifiers"
Nākošajā zīmējumā redzams pilns lietotāju tiesību izpildes laika modelis:

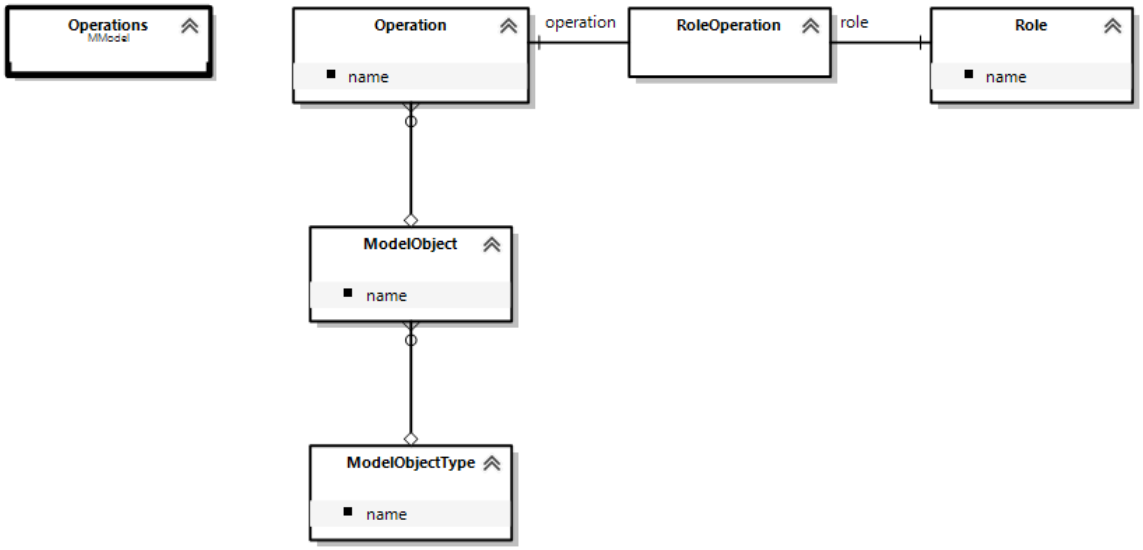



**Zīmējums 7**

Katrai entītijai ar tipu AOperationSimple no šī modeļa atbilst entītija ObjectOperation, un katrai entītijai ar tipu AuthorizationGroup, atbilst entītija GroupOperation loģiskajā lietotāju tiesību modelī. Katra šāda entītija nozīmē ierakstu lietotāju tiesību datubāzē.
Izpildes laikā ir jāpārbauda, vai lietotājam ir tiesības izpildīt atbilstošās vienkāršās un grupu operācijas, apstaigājot operāciju koku.

# 4. Tiesību modelis

Ņemot par pamatu loģisko modeli, definēsim modeli, kurš tiks izmantots lietotāju tiesību atbalsta koda un lietotāju tiesību datubāzes aizpildījuma ģenerācijai. Šis modelis ļauj definēt lietotāju lomas un operācijas un pieļauj iespēju pieejas tiesības operācijām definēt gan datubāzē – t. i. iespēju regulēt operāciju pieejas tiesības lietojumprogrammas izpildes laikā, gan lomu tiesības izpildīt operācijas ieģenerēt lietojumprogrammas izejas kodā. Sīkāk par to, ka lietotāju tiesības tiks glabātas datubāzē ir aprakstīts [6].




**Zīmējums 8**

## 4.1. ModelObjectType

**ModelObjectType** reprezentē modeļa objekta tipu

| Vārds | Datu tips | Apraksts |
|---|---|---|
| name | string | Modeļa objekta tipu (tipa vārds) |

## 4.2. ModelObject

**ModelObject** reprezentē modeļa objektu (modeļa objekta tipa instanci)

| Vārds | Datu tips | Apraksts |
|---|---|---|
| name | string | Modeļa objekta vārds |

## 4.3. Operation

**Operation** reprezentē modeļa operāciju

| Vārds | Datu tips | Apraksts |
|---|---|---|
| name | string | Operācijas vārds |

## 4.4. Role

**Role** reprezentē lomu

| **Vārds** | **Datu tips** | **Apraksts** |
|---|---|---|
| name | string | Lomas vārds |

## 4.5. RoleOperation

**Role** sasaista operācijas ar Lomām.

| **Vārds** | **Datu tips** | **Apraksts** |
|---|---|---|
| role | FK | Loma |
| operation | FK | Operācija |

# 5. Rezultāti

Aktivitātes ietvaros ir izstrādāts Loģiskais lietotoāju tiesību modelis, izpildes laika modelis, tiesību modelis, kā arī veikta tiesību modeļa automātiskās ražošanas transformācijas pārveide.

# 6. Literatūras saraksts

[1] Role Based Access Control (RBAC) and Role Based Security.
http://csrc.nist.gov/groups/SNS/rbac/
[2] Attribute Based Access Control (ABAC) – overview.
http://csrc.nist.gov/projects/abac/
[3] Claims Based Authorization Using WIF http://msdn.microsoft.com/en-us/library/hh545448%28v=vs.120%29.aspx
[4] Don’t Do Role-Based Authorization Checks; Do Activity-Based Checks
http://lostechies.com/derickbailey/2011/05/24/dont-do-role-based-authorization-checks-do-activity-based-checks/
[5] 3.8.1. "WCF biznesa saskarnes meta modelis" progresa pārskats
[6] 3.9.4 "Lietotāju tiesību modelēšanas risinājumu izpēte" progresa pārskats