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. 3.10.2 "Modeļu pārvaldība (apakšmodeļu izmantošana)" progresa pārskats

Pārskats Nr. 36 par periodu no 2015.gada 1.janvāra līdz 2015.gada 30.jūnijam.

SATURS

1. Kopsavilkums

Pārskata periodā (2015-01-01 – 2015-06-30) projekta „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem” aktivitātes "Modeļu pārvaldība (apakšmodeļu izmantošana)" ietvaros veikti šādi darbi:

1. Meta modeļu mantošanas izpēte un analīze, kas ietver meta modeļa objektu Mobject, Mmattributed, MMAttribute, Mcomplex, MModel, Mpart, MAttribute, MLink, Mreferencable, Mmodeling, MClass, MEntity, MCluster analīze un mantošanas likumu izstrāde.
2. Meta modeļu mantošanas realizācijas izstrāde.
3. Aktivitātes pētnieciskā darbība apspriesta ik nedēļas projekta semināros.

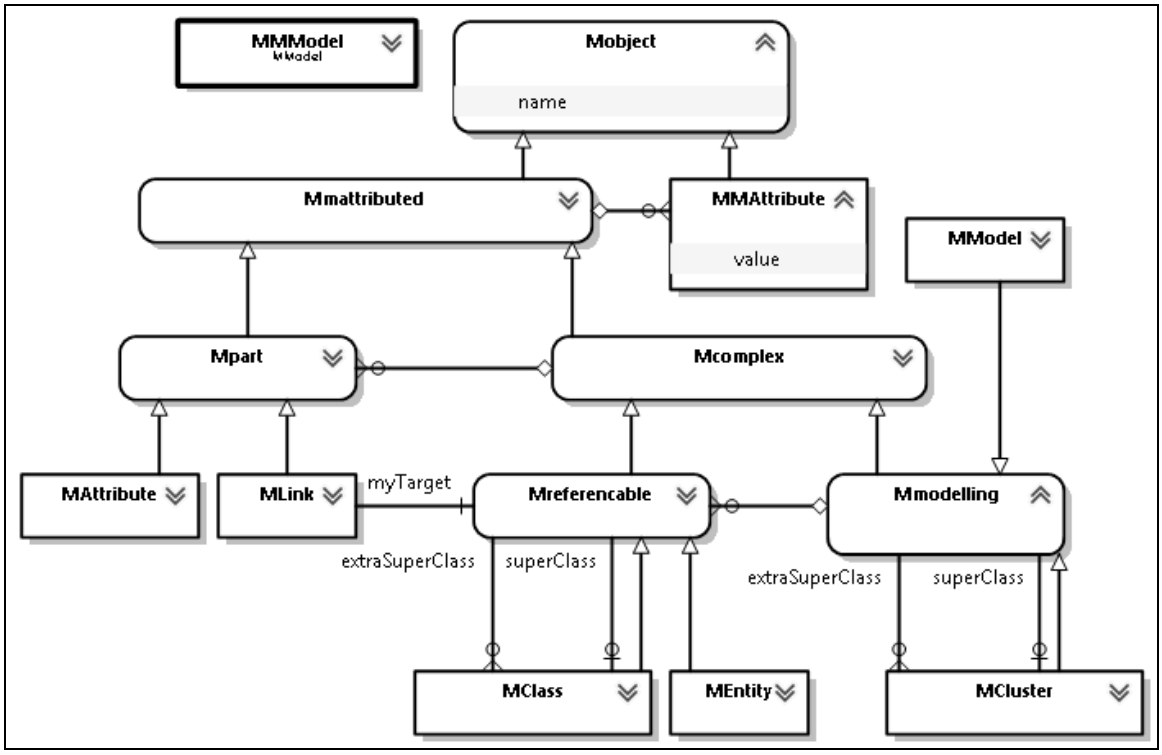
2. Ievads

Šis pārskats ir veltīts projekta apakšaktivitātes Nr. 3.10.2 " Modeļu pārvaldība (apakšmodeļu izmantošana)" progresa ietvaros veiktajiem darbiem.

Pārskatā aprakstīti Meta modeļu mantošanas likumi, kas ietver mantošanas līkumus bāzes meta modeļu objektiem – MModel, MCluster, MClass, MEntity, kā arī dots vispārīgs apraksts meta modeļu realizācijai.

3. Meta modeļu mantošana

Meta modeļu mantošanas apspriešanai ir būtiski skaidri stādīties priekšā [1] ieviesto meta meta modeli, kura diagramma atkārtota zemāk ievietotajā zīmējumā.



Zīmējums 1. Meta meta modelis.

Šajā nodaļā aplūkosim mantošanu loģiskā līmenī, t.i. aprakstīsim, kāds meta modelis rodas mantošanas ceļā. Aprakstīšanas gaitā tiks precizēti mantošanas pielietošanas ierobežojumi.

Apraksts tiks organizēts pa meta meta objektu tipiem. Tā kā patstāvīgie objekti ir *MComplex* paveidi, tad tie arī tiks aplūkoti.

Papildus grūtības rada daudzkāršā mantošana (ne velti nedz C# nedz java neatbalsta daudzkāršo mantošanu). Tādēļ apraksts tiks veidots 2 soļos - vispirms vienkāršās mantošanas gadījums (A:B) un tad papildus informācija daudzkāršās mantošanas ($A:B_1,B_2,...B_n$) atbalstam.

Tātad tālāk seko apraksts, sadalīts pa bāzes meta modeļa tipiem.

3.1. Meta modeļa objekts (MModel)

3.1.1. B.MModel

Tā kā gala meta modelī drīkst būt tieši viens meta modeļa objekts, tad bāzes meta modeļa objekts tiek pārveidots par meta klasteri (MCluster), saglabājot visu pārējo informāciju (atribūtu, meta atribūti un norādes). Jaunizveidotais klasteris tipiski tiek pievienots A.MModel pa norādi *superClass*. Ja A.MModel jau ir izmantota *superClass*, tad tiek izveidota jauna norāde **extraSuperClass**.

Ir aizliegts meta modeli iekļaut vairāk ka vienu reizi.

3.1.2. B_1,B_2,...B_n MModel

Daudzkāršās mantošanas gadījums meta modeļa objektiem tiek realizēts kā vienkāršo mantošanu virkne ar B_i.MModel atbilstoši norādītajai bāzes meta modeļu virknei.

3.2. Meta klasters (MCluster)

Meta klasteri patiesībā ir agrāko meta modeļa objektu attēli (elementārā meta modelī tie nav atļauti) un ir savstarpēji neatkarīgi. Tādēļ tie nonāk gala meta modelī bez izmaiņām gan vienkāršā gan daudzkāršā mantošanā.

3.3. Meta klase (MClass)

Ļoti svarīgi apzināties, ka meta klase tiek viennozīmīgi identificēta ar vārdu. Tas nozīmē, ka visi simboli (gan klases gan klases atsauces), kas apzīmē klasi un ir ar vienu vārdu, patiesībā nozīmē vienu un to pašu klasi. Saturīgi tas nozīmē, ka meta klasi nav iespējams papildināt ar papildus sastāvdaļām apakšmetamodeļos.

3.3.1. B.MClass

Tā kā A modelī meta klases vārds var tikt izmantots dažādi vai pat vispār netikt izmantots, tad jāaplūko gadījumi:

- tipiskais gadījums ir, ka A meta modelī ir atsauce (skat. [2]) uz klasi ar doto vārdu A.MClassPlaceHolder. Šajā gadījumā B.MClass nonāk gala meta modelī. Visas atsauces uz A.MClassPlaceHolder (var būt tikai A meta modelī) tiek pārtaisītas par atsucēm uz gala meta modelī iekļauto B.MClass. Savukart A.MClassPlaceHolder tiek likvidēts.
- A meta modelis nesatur *Mreferencable* ar B.MClass vārdu. Šajā gadījumā B.MClass nonāk gala meta modelī.
- A meta modelis satur *Mreferencable* ar B.MClass vārdu. Tā ir **nepieļaujama situācija**, un gala meta modeli nav iespējams izveidot.

3.3.2. $B_1,B_2,...B_n$ MClass

Ja meta klase ar doto vārdu sastopama tieši vienā bāzes meta modelī B_k, tad mantošana notiek atbilstoši iepriekš aprakstītajai vienkāršās mantošanas kārtulai.

Ja meta klase ar doto vārdu vairāk kā vienā bāzes meta modelī, tad tā ir **nepieļaujama situācija** un tiek uzskatīta par kļūdu. Šis ierobežojums lielā mērā ir saistīts ar tālāk plānoto realizāciju vidē (.NET), kas neatbalsta daudzkāršo mantošanu.

3.4. Meta entītija (MEntity)

Atšķirībā no meta klases meta entītija dažādos meta modeļos var atšķirties ar sastāvdaļu repertuāru (tehniskais iemesls ir tas, ka meta meta modeļa nozīmē meta entītija nav tālāk mantojama).

3.4.1. B.MEntity

Tā kā A modelī meta entītijas vārds var tikt izmantots dažādi vai pat vispār netikt izmantots, tad jāaplūko gadījumi:

- A meta modelis nesatur *Mreferencable* ar B.MEntity vārdu. Šajā gadījumā B.MEntity nonāk gala meta modelī.
- A meta modelis satur *MClass* ar B.MEntity vārdu. Tā ir **nepieļaujama situācija,** un gala meta modeli nav iespējams izveidot.
- A meta modelis satur *MEntity* ar B.MEntity vārdu Šajā gadījumā B.MEntity nonāk gala meta modelī pārveidota par meta klasi. Savukārt A.MEntity tiek pievienota atsauce *superClass* (ja aizņemta, tad jauns *extraSuperClass* eksemplārs) uz jaunizveidoto meta klasi.

3.4.2. $B_1,B_2,...B_n$ MEntity

Ja meta entītija ar doto vārdu sastopama tieši vienā bāzes meta modelī B_k, tad mantošana notiek atbilstoši iepriekš aprakstītajai vienkāršās mantošanas kārtulai.

Ja meta entītija ar doto vārdu vairāk kā vienā bāzes meta modelī, tad tā ir **nepieļaujama situācija** un tiek uzskatīta par kļūdu.

4. Meta modeļu realizācija

Iepriekšējā nodaļā aprakstītie mantošanas likumi jau ir samērā sarežģīti loģiskā līmeni. Realizācija kļūst ļoti komplicēta, jo no praktiskās pielietošanas viedokļa ir svarīgi iegūt iespēju mantot arī ar meta modeli asociētu programmas kodu.

Savulaik [3] tika aprakstīts D-modelis kā struktūra, kas ļauj samērā ērti manipulēt ar modeļa datiem. Dabisks šīs idejas turpinājums ir meta modelim atbilstoša dinamiskā bibliotēka (DLL), kas sevī realizē šī meta modeļa D-modelisku interpretāciju.

Tika izstrādāta speciāla T4 transformācija (DModel.ttinclude), kas no meta modeļa apraksta veido C# kodu D-modeļa realizācijai. Praktiski tas nozīmē specializētas apakšklases atbilstoši meta modeļa objektiem, kas realizē DModel programmatūras izmantošanu.

Atsevišķi ņemtam meta modelim šī realizācija būtu samērā vienkārša. Ņemot vērā mantošanu, kas savukārt nozīmē atsauces uz bāzes meta modeļu DLLiem (patvaļīgā dziļumā), tas nav pilnīgi automatizējams darbs. Tika izveidots risinājums (solution) MetaModels, kurā katru meta modeli pārstāv savs projekts. Šie meta modeļu projekti tiek veidoti pēc vienotas shēmas, izmantojot iepriekš minēto T4 transformāciju.

Jauna meta modeļu projekta veidošanas secība:

- atvērt MetaModels.sln
- izveidot jaunu projektu ar tipu "Class Library (Portable)" un atbilstoši konkrētā pielietojuma mērķiem izvēlēties atbalstītās vides
- projekta īpašībās norādīt Build.Output Path ..\..__DLL\
- pievienot atsauci uz meta modeļa failu (..\.._\<meta modelis>.mmedus)
- izveidot T4 transformāciju failu ar patvaļīgu vārdu (piemēram, mans.tt), kurš satur
    - `<#@ include file="./../_TT/fromEd.ttinclude" #>`
    - `<# Dmake("<meta modelis>"); #>`
- pievienot atsauces uz dinamisko bibliotēku MEDUS.dll
- pievienot atsauces uz dinamiskajām bibliotēkām, kas reprezentē bāzes meta modeļus

5. Rezultāti

Aktivitātes ietvaros ir izstrādāti meta modeļu mantošanas likumi, un ir veikta meta modeļu mantošanas realizācija.

6. Literatūras saraksts

1. Nr.1.1 „Meta metamodeļu izpēte” progresa pārskats
2. Nr.2.1. „Metamodeļu grafiskā redaktora izstrāde” progresa pārskats
3. Nr. 2.3.2 „Modeļa importa izveide (T2M transformāciju atbalsts)” progresa pārskats