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.3.1 "Izstrādes vides (.NET/Mono, T4/neatkarīgs ģenerācijas rīks) iespaida izpēte" progresa pārskats

Pārskats Nr. 12 par periodu no 2014.gada 1.janvāra līdz 2014.gada 30.jūnijam.

SATURS

1. Kopsavilkums....4
2. Ievads....7
3. „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" izstrāde....8
3.1. MS .NET tehnoloģija....8
3.2. Projekti-sastāvdaļas (tipi)....8
3.3. Koda ģenerācija: T4....8
3.4. Modelis koda ģenerācijā....9
4. Cita .NET realizācija – Mono....10
4.1. Savietojamība ar MS .NET....10
4.2. MonoDevelop izstrādes vide....10
4.2.1. Automatizētās testēšanas līdzekļi....11
4.3. Rīks "MoMA – Mono Migration Analyzer"....11
5. Migrācija no MS .NET uz Mono vidi....12
5.1. Migrācijā sastaptās problēmas....12
5.1.1. .NET 2.0 un .NET 4.0 rīki Mono vidē....12
5.1.2. MS Visual Studio dienesta faili MonoDevelop vidē....12
5.1.3. MS .NET projektu tipi MonoDevelop vidē....12
5.2. Bibliotēku pieejamība....13
5.2.1. Entity Framework bibliotēka....13
5.3. T4 ģenerācija Mono vidē....13
5.3.1. Atšķirības un trūkumi....14
5.3.2. Paštaisītais ģenerators....14
6. Licenču jautājumi....15
6.1. Web-aplikācijas darbināšana Mono vidē....15
6.2. .NET bibliotēku licence....15
6.3. .NET Framework Redistributable licence....15
7. Secinājumi....16
8. Literatūras saraksts....17

1. Kopsavilkums

Pārskata periodā (2014-01-01 – 2014-06-30.) projekta „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" aktivitātes Nr.3.1 "Izstrādes vides (.NET/Mono, T4/neatkarīgs ģenerācijas rīks) iespaida izpēte" ietvaros veikti šādi darbi:

1. Apzinātas kādas .NET tehnoloģijas būs nepieciešamas projekta izstrādē;
2. Ģenerācijas izpēte Microsoft Visual Studio, kas ietver:
    1. T4 tehnoloģijas izpēti un izmēģināšanu, tas ir:
        1. Paraugfailu iekļaušanas (.ttinclude) iespēju izpēte;
        2. Daudzu failu ražošana no viena parauga: manuālās realizācijas iespējas un Entity Framework bibliotēkas izmantošana;
    2. T4 ģenerācijā izmantojamo kopējo funkciju un to infrastruktūras izstrādi;
    3. T4 paraugfailu infrastruktūras izstrādi, maksimāli norobežojot tehnoloģisko un konkrētā pielietojuma daļas;
    4. T4 izpildes laikā darbināmās ģenerācijas izmēģināšana;
3. Vides izveide Mono darbināšanai, tas ir:
    1. Parauga operētājsistēmas izvēle - OpenSuse Linux 13.01;
    2. Mono un MonoDevelop un saistīto pakešu instalēšana (mono-project.com un OpenSuse iebūvētā variantu izmēģināšana);
4. Mono vides izpēte projekta vajadzībām;
    1. Savietojamības starp Mono un Microsoft .NET izpēte, kas ietver:
        1. Mono dokumentācijas pētīšanu;
        2. Imēģinājuma projektu darbināšanu starp vidēm;
5. T4 ģenerācijas analīze Mono vidē, kas ietver:
    1. T4 ģenerācijas darbināšanas iespēju izpēti;
    2. T4 paraugfailu savietojamības konfliktu risināšanu (starp Microsoft un Mono vidēm);
    3. MonoDevelop izstrādes vides pielāgošanu regulārai T4 ģenerācijas darbināšanai;
6. Automatizētās testēšanas iespējas izpēte Mono vidē, tas ir:
    1. Microsoft UnitTest un NUnit atšķirību izpēte;
    2. Automatizētās testēšanas iedarbināšanas izpēte iekš MonoDevelop priekš .NET 4.0 vides;
    3. Automatizētās testēšanas ceļu izpēte bez MonoDevelop infrastruktūras iesaistīšanas;
7. Rīka "MoMA – Mono Migration Analyzer" analīze;
8. Migrācijas no Microsoft Visual Studio .NET uz Mono vidi izpēte, kas ietver:
    1. Microsoft .NET projektu rediģēšanu MonoDevelop izstrādes vidē, savietojamības konfliktu risināšanu;
    2. Jaunu projektu izstrādi Mono vidē (atbilstoši MS .NET projektu tipiem un izmantojot jau izstrādātus izejas koda failus):
        1. Referencēto bibliotēku piemeklēšana Mono vidē;
        2. Izejas koda failu pielāgošana izmantošanai gan Microsoft, gan Mono .NET vidēs;

3. Entity Framework 6.1 bibliotēkas izmantošanas nepieciešamības izpēte;

9. Licenču jautājumi, kas rodas Microsoft vidē, izstrādāto programmatūru pārnesot uz Mono.

2. Ievads

Šis pārskats ir veltīts projekta apakšaktivitātes Nr.3.1. "Izstrādes vides (.NET/Mono, T4/neatkarīgs ģenerācijas rīks) iespaida izpēte" ietvaros veiktajiem pētījumiem.

Projektu plānots izstrādāt Microsoft Visual Studio vidē, tajā pat laikā tika apskatīta iespējamā alternatīva - Mono, jo Microsoft Visual Studio ierobežo izstrādātās programmas pielietošanas vidi ar Windows platformu, kas dažreiz varētu konfliktēt ar reālām prasībām. Līdz ar to sākotnēji tika izanalizēts, kādas .NET tehnoloģijas būs nepieciešams projekta izstrādē, īpašu uzmanību veltot T4 koda ģenerācijai.

Apakšaktivitātes galvenais pētījuma objekts ir Mono, kas ir OpenSource projekts, kas ļauj realizēt .NET tehnoloģiju uz dažādām platformām [1]. Mono ļauj izpildīt programmas, kas tika izstrādātas ne-Mono vidē – ar Microsoft .NET. Tika izpētītas Mono vides iespējas, tai skaitā automatizētās testēšanas iespējas, rīks MoMA – Mono Migration Analyzer.

Liela uzmanība tika veltīta migrācijai no Microsoft Visual Studio vides uz Mono, tika identificētas konkrētas problēmas.

Papildus tika analizēta T4 ģenerācija Mono vidē un tās iespējamās alternatīvas, kā arī izpētīts licencēšanas jautājumi, kas rodas Microsoft vidē, izstrādāto programmatūru pārnesot uz Mono.

3. „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" izstrāde

3.1. MS .NET tehnoloģija

Par projekta „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" izstrādes pamatvidi tika izvēlēta Microsoft .NET vide (ar Microsoft SQL Server datu bāzi kā datu glabātuvi).

3.2. Projekti-sastāvdaļas (tipi)

„Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" risinājuma pakete sastāv no vairākiem dažādu tipu projektiem.

Projekta tips	Mērķis
Class Library	Ir parastās klašu bibliotēkas, kas piedāvā funkcionalitāti serveru vai klientu programmām. Bieži vien atsaucās uz Microsoft bibliotēkām (piemēram, uz Entity Framework, darbam ar datu bāzi).
.NET Portable	Klašu bibliotēkas tips, kuru atsauces ir ierobežotas ar .NET pārnesāmo bibliotēku kopu: tā ir darbināma gan Windows 7/8.*, gan Windows Mobile. Tas ierobežo, kādas vēl bibliotēkas drīkst piesaistīt šim projektam: tām visām jābūt atkarīgām tikai no pārnesamām bibliotēkām. (Piem., Entity Framework nav tāda).
Unit Test	Testu projekts: bibliotēku funkciju automatizētai testēšanai. MS Visual Studio satur infrastruktūru šo testu laišanai un rezultātu apkopošanai.

3.3. Koda ģenerācija: T4

Koda ģenerēšanai risinājumā „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" izmanto T4 tehnoloģiju, kas ir iebūvēta iekš Microsoft Visual Studio, kā arī tās pielietošanas funkcijas. T4 ir izcila ar to, ka:

- ģenerējamo bloku manipulācijas tiek rakstītas ar C# programmēšanas valodu (vai citu valodu, ko atbalsta Microsoft Visual Studio);
- ļoti pārskatāmi apraksta paraugu: bez īpašām pūlēm no tā var redzēt, kā izskatīsies rezultāts.

T4 tehnoloģija ļauj izmantot klašu bibliotēkas, kas tiek automātiski piesaistītas, kad tiek izpildīta ģenerācija.

Paraugfailu formāts pieļauj iespēju iekļaut paraugfailus vienu otrā. Mūsu risinājumā „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" šī iespēja tiek plaši izmantota: pamatfunkcionalitāte koda ģenerācijai tiek sadalīta pa kopējiem failiem. Paraugfailu ģenerācija tomēr ir iekļauta projektu/risinājumu struktūrā, līdz ar to failos var atsaukties uz iekšējiem MS Visual Studio mainīgajiem: *ProjectDir*, *SolutionDir* utt.

Koda ģenerācija neietilpst kompilācijas procesā. Microsoft Visual Studio piedāvā saskarni, kā to izsaukt manuāli.

3.4. Modelis koda ģenerācijā

Lai iedarbinātu koda ģenerāciju, ir nepieciešams modelis: modeļa dati (mūsu gadījumā – atsevišķs .xml fails) un ielasīšanas rīks, kas saliks modeļa datus pareizās struktūrās ērtai izmantošanai (mūsu gadījumā - ar .NET tehnoloģiju izstrādātā klašu bibliotēka .dll, kurā ir iebūvēta gan ielasīšana no modeļa faila, gan metamodelis, kas definē struktūras modeļa datiem).

4. Cita .NET realizācija – Mono

MS .NET izvēle ļauj izmantot visus servisus un priekšrocības, ko izstrādātājiem piedāvā Microsoft Visual Studio izstrādes programmatūra, nodrošinot ātru un ērtu programmēšanu. Bet arī ierobežo izstrādātās programmas pielietošanas vidi ar Windows platformu, kas dažreiz varētu konfliktēt ar reālām prasībām.
Tomēr ar Windows platformu .NET tehnoloģija neierobežojas.

Mono – ir OpenSource projekts, kas ļauj realizēt .NET tehnoloģiju uz dažādām platformām [1]. Jo vairāk: tas ļauj izstrādāt programmas uz vienas platformas un izpildīt uz citas.

Mono ļauj izpildīt programmas, kas tika izstrādātas ne-Mono vidē – ar Microsoft .NET. Diemžēl, ar dažiem izņēmumiem.

4.1. Savietojamība ar MS .NET

Oficiāli Mono atbalsta sekojošas lietas no .NET tehnoloģijas[2]:

- C# 5.0
- LINQ
- Asp.NET MVC 3 (un daļēji – MVC 4, bez asinhronitātes atbalsta)
- Entity Framework 6.0
- WCF (daļēji)

Neatbalsta (un arī neplāno):

- WCF
- WWF

4.2. MonoDevelop izstrādes vide

Mono vidē pastāv arī Microsoft Visual Studio analogi, kas ļauj ne-Windows platformās ērtā vizualizētā veidā izstrādāt programmas. Linux vidē tas ir MonoDevelop studija.
Izskatā MonoDevelop ir ļoti līdzīgs Microsoft Visual Studio. Eksperimentu gaitā pamanītās būtiskas atšķirības ir sekojošas:

- var strādāt ar Microsoft Visual Studio darba failiem (risinājuma un projektu), bet ne no pēdējās Visual Studio versijas (VS 2013); tomēr pastāv veids, ka to apiet;

- ģenerācijai no paraugfailiem nav funkcionālā atbalsta MonoDevelop saskarnē. To var organizēt, piekonfigurējot saskarni, bet galu galā tas reducējas uz neatkarīgās konsoles programmas pielietošanu.

MonoDevelop visplašāk uzturamā programmēšanas valoda ir C# un C. Ierobežoti ir pieļaujamas C++ un Visual Basic.

4.2.1. Automatizētās testēšanas līdzekļi

Izstrādes gaitā rodas nepieciešamība notestēt jau gatavus gabalus. Tā kā vēl izstrādē nav iesaistīta klienta vizuālā saskarne, tad pilnīgi pietiekami sagatavot un laist batch-testus – programmas, kas izsauc izstrādātās funkcijas un automātiski pārbauda rezultātus.
Microsoft Visual studio piedāvā šim mērķim jau gatavu infrastruktūru: Unit Test projekta sagatavi un studijā iebūvēto funkcionalitāti, kā šādu projektu iedarbināt un pēc tam apkopot rezultātus.

Testēšanai iekš MonoDevelop speciālā atbalsta nav. Bet ir NUnit testu projekts (UnitTest analogs, kas pastāv visās vidēs). Tā testu iezīmēšana nesakrīt pilnībā ar MS Unit Test testiem, bet atšķirības ir minimālas (atribūtu nosaukumi un cita izmantotā bibliotēka). MonoDevelop izstrādes vidē ir arī iebūvēta infrastruktūra, kas ļauj darbināt testus un apkopot rezultātus.

4.3. Rīks “MoMA – Mono Migration Analyzer”

Mono vidē pastāv speciāls rīks – palīgs .NET aplikāciju migrēšanai no Microsoft uz Mono vidi. Šis rīks .NET kompilētajam produktam (.dll vai .exe) noskaidro, kādas būs problēmas tā pārcelšanā uz Mono vidi. Diemžēl, šis rīks ir pamatīgi vecs (pēdējā Mono versija, pret kuru tas salīdzina, ir 2.8; aktuālā versija ir 3.4.0), tāpēc mūsu gadījumā tas nav pielietojams.

5. Migrācija no MS .NET uz Mono vidi

Mono vide ļauj arī kompilēt kodu, kas ir izstrādāts Microsoft .NET videi (ar dažiem izņēmumiem). Līdz ar to, ja ir zināms, ka projekts tiks kompilēts arī Mono vidē, ir vērts izstrādāt kodu, maksimāli derīgu abās vidēs.

5.1. Migrācijā sastaptās problēmas

5.1.1. .NET 2.0 un .NET 4.0 rīki Mono vidē

Mono vide atbalsta gan .NET 2.0, gan .NET 4.0 aplikāciju laišanu un izstrādi. Diemžēl katrai no šīm vides versijām pastāv atsevišķa dienesta rīku kopa (vides darbināšana, kompilatori, testēšanas saimniecība, web-serveri utt.) un savs bibliotēku komplekss. Pēc noklusēšanas visur tiek izmantota versija 2.0

MonoDevelop spēj pārslēgt kompilācijas procesu atkarībā no tā, kāda .NET versija ir norādīta projekta failā. Bet pārējos procesus (ģenerācijas/testu laišanu, web-servera darbināšanu) ir jāpielāgo pašam, laižot .NET 4.0 versijas dienesta rīkus ārpus MonoDevelop.

Viena no kļūdām, kas signalizē par nepareizas versijas izmantošanu kāda dienesta rīka darbināšanā ir:
TypeInitializerException: IConfigurationSectionHandler:Create
MissingMethodException

5.1.2. MS Visual Studio dienesta faili MonoDevelop vidē

MonoDevelop var strādāt ar MS Visual Studio risinājumu/projektu failiem, bet tikai līdz Visual Studio 2012 versijai (ieskaitot). Tā kā tiek izmantota Visual Studio 2013, MonoDevelop ar šiem failiem nestrādā.
Savietojamības konflikts rodas no risinājuma atribūta "*ToolsVersion*" vērtības: Visual Studio 2013 ieliek "12.0", bet MonoDevelop sagaida "4.0". Ja nomainīt, pārējo formātu MonoDevelop akceptē.
Tika pieņemts lēmums, ka tiks taisīts jauns risinājums un jauni projekti, kuri tiks piepildīti ar izejas kodu failiem no MS .NET projektiem.

5.1.3. MS .NET projektu tipi MonoDevelop vidē.

Pārsvarā MonoDevelop vide atbalsta tos projektu tipus, kas tiek izmantoti risinājumā „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" izstrādē.
Eksperimentu gaitā parādījās dažas īpatnības darbam ar risinājumā „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" izmantotiem projektu tipiem:

.NET Portable	Šāds tips arī pastāv Mono vidē, MonoDevelop pārzin tādu projektu sagatavi un uztaisa pēc tās projektu. Diemžēl, .NET Portable referenču kopa Mono vidē ir instalējama atsevišķi. Mūsu gadījumā to neizdevās iedarbināt. Attiecīgais MEDUS projekts tika migrēts kā vienkārša klašu bibliotēka.
NUnit (analogs Unit Test)	Izrādījās, ka NUnit testēšana iekš MonoDevelop ir stingri orientēta uz .NET 2.0 vides programmām: mēģinot palaist .NET 4.0 (un augstāk) testus, parādās kļūda. .NET 4.0 testus var izstrādāt ar MonoDevelop, bet tad tie ir jālaiž komandrindā ar komandu: *nunit-console4 -run=<testa nosaukums> <NUnit .dll nosaukums>* *nunit-console4* iedarbina infrastruktūru, kas ir domāta .NET 4.0 versijai, un testēšana notiek. Bet parādās problēmas, kas saistītas ar kļūdām testējamās bibliotēkās, ko atrod paši testi: kļūdu informācijas izdrukāšanas formāts ir neērts, un atkļūdošana ir apgrūtināta. Galu galā testēšana tika realizēta ar neatkarīgu konsoles aplikāciju, kurā ievietoja Unit Test testu ķermeni.

5.2. Bibliotēku pieejamība

Risinājumā „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" ir nepieciešams izmantot pāris bibliotēkas, kurām Mono vidē nav analoga: Dynamic LINQ (dinamisko vaicājumu atbalsts) un Entity Framework 6.1. (savienojums ar datu bāzēm; jaunākā, kas ir pieejama iekš Mono, ir Entity Framework 6.0). Šīs bibliotēkas pa tiešo iekopētas no MS .NET vides (tā kā Mono atbalsta arī MS .NET kompilēto programmu izpildi). Pārnešana ir veiksmīga.

5.2.1. Entity Framework bibliotēka

„Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" risinājumam ir nepieciešama tieši Entity Framework 6.1 versija, jo tas pēc būtības izmanto jaunāko funkcionalitāti.
Entity Framework jaunākajā versijā parādījās iespēja lietot *ToString()* funkciju LINQ pieprasījumos: jau datu bāzēs pieprasījuma līmenī jebkuras kolonas vērtību var pārtaisīt par simbolu virkni un izmantot to filtrācijas nosacījumos, kas operē ar simbolu virknēm (āgrāk tas nebija iespējams). Uz šīs iespējas ir bāzēta entītiju reprezentatīvās vērtības vispārināšana.

5.3. T4 ģenerācija Mono vidē

MonoDevelop studija satur arī rīku, kas ļauj ģenerēt kodu T4 paraugiem ar gandrīz tādu paši sintaksi, kā iekš Microsoft Visual Studio. Diemžēl pašā MonoDevelop studijā nav

iebūvētā atbalsta, kas ļautu iedarbināt ģenerāciju no paraugiem. (MonoDevelop apraksts apgalvo, ka T4 parauga faila saglabāšana palaiž ģenerāciju, bet vairāku eksperimentu rezultātā tika noskaidrots, ka MonoDevelop to dara tikai vienu reizi, sākot no palaišanas brīža: ja grib atkārtoti pārģenerēt paraugu, ir jāpārstartē studija.) MonoDevelop ļauj sakonfigurēt savu saskarni lietotāja batch komandas izpildei: to ir ieteicams izmantot, lai ieviestu ģenerāciju no parauga (to var iebūvēt arī pirmskompilācijas procesā). Bet tas nozīmē, ka tik un tā ģenerācijas procedūra reducējas uz konsoles rīka palaišanu:

mono /usr/lib/monodevelop/AddIns/MonoDevelop.TextTemplating/TextTransform.exe
--out=<paraugfails>
<ģenerētā faila vārds>

Līdz ar to nav nekādu priekšrocību izmantot tieši T4 ģenerāciju, ja nav nekādu citu ierobežojošu faktoru.

5.3.1. Atšķirības un trūkumi

Mono ģenerācijas rīkam ir daži ierobežojumi, salīdzinot ar Microsoft rīku:

- "include" direktīva nedrīkst saturēt citus atribūtus, izņemot "file";
- tā kā Mono vidē ģenerācija faktiski notiek ārpus studijas, paraugu faili nedrīkst saturēt ceļus ar MS Visual Studio konstantēm "*ProjectDir*" un "*SolutionDir*";
- Mono vidē nevar izmantot Entity Framework mehānismu, kas ļauj ģenerēt no viena paraugu faila vairākus koda failus. Ir jārealizē daudzu failu saglabāšana pašam.

Ar vairāku failu ražošanu ir viena nianse: Entity Framework nodrošina ģenerācijai arī automātisku jaunuzražoto koda failu piesaistīšanu projektam. Diemžēl, projekta satura manipulēšana Microsoft .NET vidē atrodas EnvDTE.dll bibliotēkā, kas nav Mono vidē. Tāpēc vairāku failu ražošana no viena parauga būtu jārealizē atsevišķi Microsoft un atsevišķi Mono vidēm. Cits risinājums ir – nokopēt šo bibliotēku no Microsoft .NET.

5.3.2. Paštaisītais ģenerators

MS Visual Studio ļauj izstrādāt programmas, kas iedarbina T4 ģenerāciju no parauga aplikācijai strādājot, un tādas aplikācijas strādās arī uz mašīnas, kur nav MS Visual Studio. To varētu izmantot, lai uztaisītu savu ģeneratoru, ko pēc tam laidīs ar Mono. Tas, iespējams, ļautu apiet kādas no atšķirībām ar Mono ģeneratoru.

Vēl viena iespēja uztaisīt savu ģenerāciju ir – lietojot simbolu virkņu funkcijas. Šis variants būtu pat pārskatāmāks, ja ģenerējamais kods ir neliels un daudz vairāk ir atkarīgs no sākuma datu analīzes, nekā no pašiem datiem.

6. Licenču jautājumi.

Vairākās reizēs, migrējot „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" risinājumu uz Mono vidi, problēmas ar trūkstošajām bibliotēkām tika atrisinātas ar to, ka nepieciešamas bibliotēkas tika kopētas no MS .NET vides. Tehniski šis risinājums nostrādā, jautājums – cik tas ir legāls licenču ziņā?

6.1. Web-aplikācijas darbināšana Mono vidē

ASP.NET bibliotēkas pieder Microsoft, bet ir zem OpenSource licences – kas pieļauj izmantošanu citās vidēs. Bet ASP.NET aplikācija var iekļaut ne tikai šīs bibliotēkas: var būt izmantots tas pats Entity Framework, piemēram. Šajā gadījumā Entity Framework bibliotēka parādīsies arī izplatāmajā (deployment) pakotnē.

6.2. .NET bibliotēku licence

Par piemēru, uz kura noskaidrot licenču jautājumus, tika paņemta Entity Framework bibliotēka. Šo bibliotēku uzinstalējot, parādās licence (EULA – End User License Agreement), kas nosaka, kādā veidā šo bibliotēku drīkst lietot [3]. Tā attiecas uz .NET bibliotēku (ne tikai konkrēti uz Entity Framework) un neaizliedz bibliotēkas izmantošanu Mono vidē.
Līdz ar to šīs bibliotēkas izmantošana ir pieļaujama gan izstrādē (arī mūsu migrācijas procedūrā), gan darbinot MS .NET vidē izstrādāto aplikāciju Mono vidē.

6.3. .NET Framework Redistributable licence

.NET bibliotēkas vēl ir dabūjamas ar lielā pakotnē .NET Framework Redistributable, kam arī ir sava licence, atrodama iekš [4]. Šī licence apgalvo, ka pakotne ir Microsoft operētājsistēmas produkta komponente, un lietot to var tikai licencētajā atbilstošajā operētājsistēmā (kā arī kopēt, aizvietot utt.).

.NET Framework Redistributable pakotne tomēr ir atsevišķs produkts ar savu instalāciju, tāpēc nevar teikt, ka iet runa par tām pašām atsevišķām bibliotēkām, bet tomēr tas padara jautājumu par kopēšanas pieļaujamību par līdz galam neatrisinātu.

7. Secinājumi

Pētījumu rezultātā ir secināts, ka Mono vide ir vāji piemērota lielu .NET projektu izstrādei, kas iekļauj uz modeļa balstītu automatizāciju, jo:

- Mono atbalstīto bibliotēku klāsts ir daudz šaurāks, nekā Microsoft .NET vidē, un to kopēšana no Microsoft vides, kaut arī iespējama tehniski, ir apšaubāma licencēšanas ziņā;

- Mono rīku noklusētā konfigurācija ir dzelžaini notēmēta uz .NET 2.0, un pēdējās .NET versijas aplikācijas iedarbināšana/izstrāde ir ļoti apgrūtināta;

- koda ģenerācijas atbalsts MonoDevelop vidē ir nepietiekams, lai pilnvērtīgi izmantotu modeli projekta vajadzībām.

Tādējādi projekta izstrāde iekš Mono (vai ar prasību, ka projekts tiks atkārtots vai darbināts iekš Mono) spiež atteikties no jaunākām tehnoloģijām Microsoft .NET vidē un turēties pie ļoti ierobežotās veco tehnoloģiju kopas, bet modeļa izmantošanas ziņā - gandrīz atkārtoti implementēt nopietnu daļu no tā, ko piedāvā Microsoft .NET izstrādes infrastruktūra. Līdz ar to tika izvēlēts, ka risinājums "Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem " tiks attīstīts Windows vidē.

8. Literatūras saraksts

[1] What is Mono (http://mono-project.com/What_is_Mono)
[2] Mono Compatibility (http://mono-project.com/Compatibility)
[3] Microsoft .NET Library EULA (http://www.microsoft.com/web/webpi/eula/net_library_eula_enu.htm)
[4] Microsoft .NET Redistributable EULA (http://msdn.microsoft.com/en-us/library/ms994405.aspx)