Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

VISOKOŠOLSKI STROKOVNI ŠTUDIJ
Delovna praksa

ZAKLJUČNO POROČILO

Geniusnet d.o.o.

Priimek in ime študenta: Gregor Majcen

Vpisna številka: 63070199

Čas izvajanja delovne prakse: 21.2.2011 – 24.4.2011

Mentor na fakulteti: doc. dr. Peter Peer

Mentor v podjetju: viš. Bojan Trajkovič

Datum: _____

Podpis študenta: Podpis mentorja v podjetju:

Kazalo vsebine

Kazalo slik

1. UVOD

Podjetje Geniusnet d.o.o. sodeluje z nemškim podjetjem Madgeniuses GmbH in sicer z razvojem sistema pixi*. Delovno prakso sem opravljal kot razvijalec programske opreme. Največ sem se posvetil projektu Slovar, kateri je bil tudi cilj moje delovne prakse.

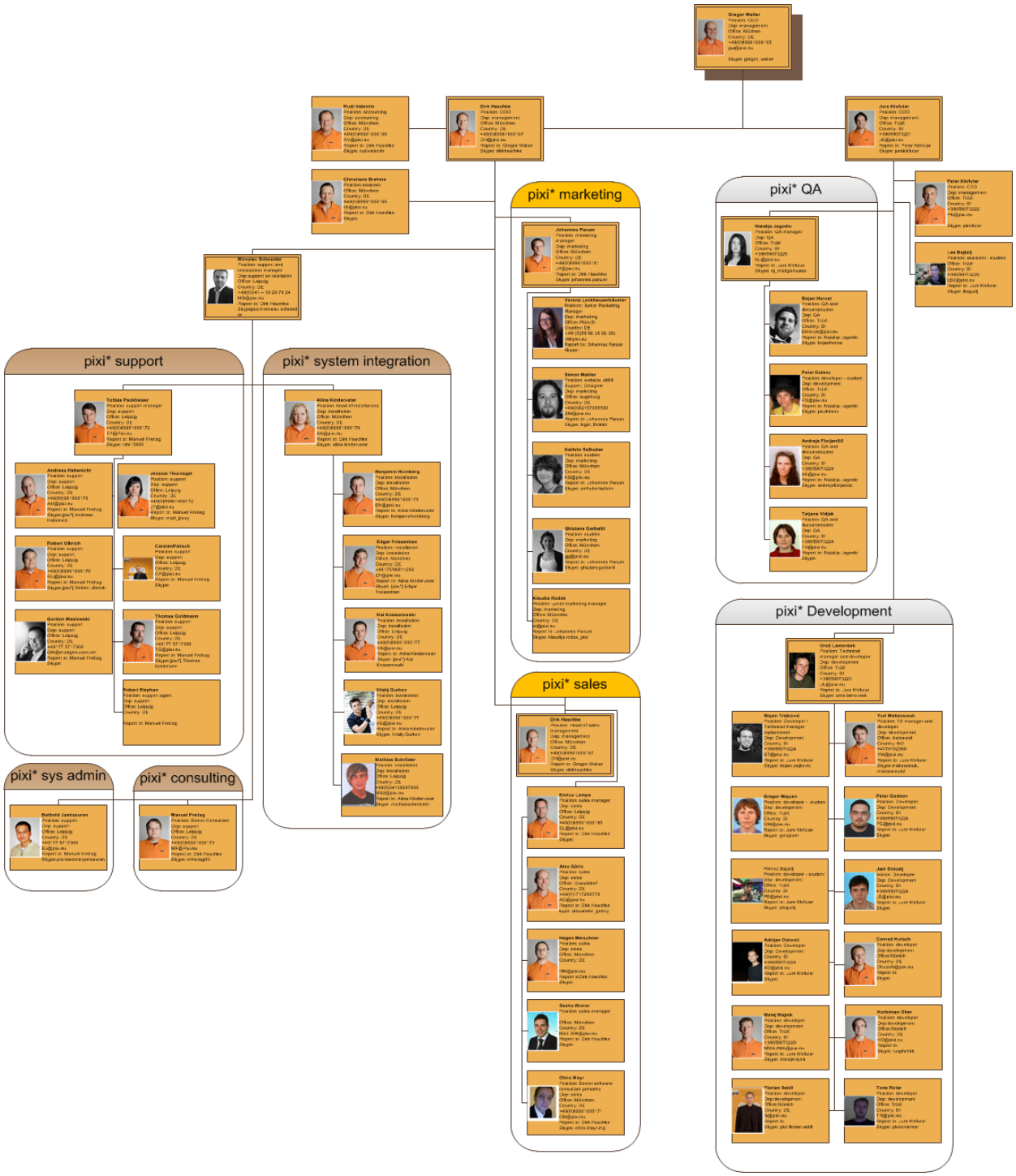
2. GLAVNI DEL

2.1 Opis podjetja

Podjetje temelji na sistemu pixi* katerega tudi sam razvija. Programska rešitev pixi* je namenjena podjetjem, ki se ukvarjajo s prodajo preko spletne trgovine ali kataloga. Najpomembnejši del sistema pixi* je optimizirano, hitro in učinkovito pošiljanje naročenih izdelkov strankam. Poleg tega pixi* pomaga voditi zalogo s čim nižjimi stroški, skrbi za pravočasno naročanje pri dobaviteljih ter v vsakem trenutku lahko preverite status naročila in ga po potrebi spreminjate. pixi* omogoča tudi avtomatsko obdelavo plačil ter ponuja vsa potrebna poročila za učinkovito vodenje podjetja.

Slika 1: pixi* paket Professional.

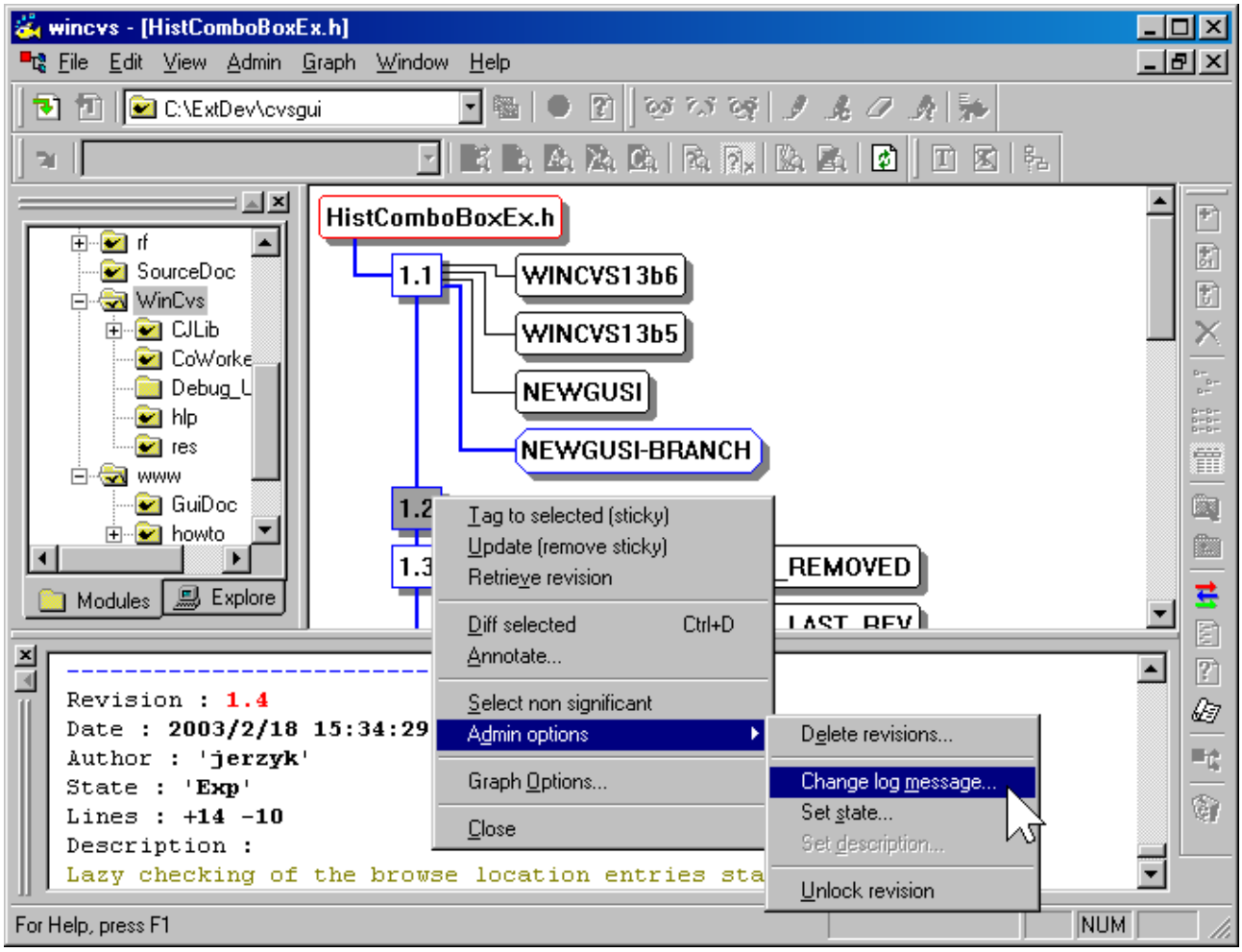
Kot večina podjetij ki se ukvarjajo z razvojem programske opreme ima tudi to podjetje več oddelkov, največ sodelovanja sem imel v QA, ki je namenjen za testiranje in zagotavljanje kakovosti, ter Development, ki je namenjen razvoju programske opreme.



Slika 2: Organizacijska struktura podjetja Madgeniuses GmbH in GeniusNet d.o.o..

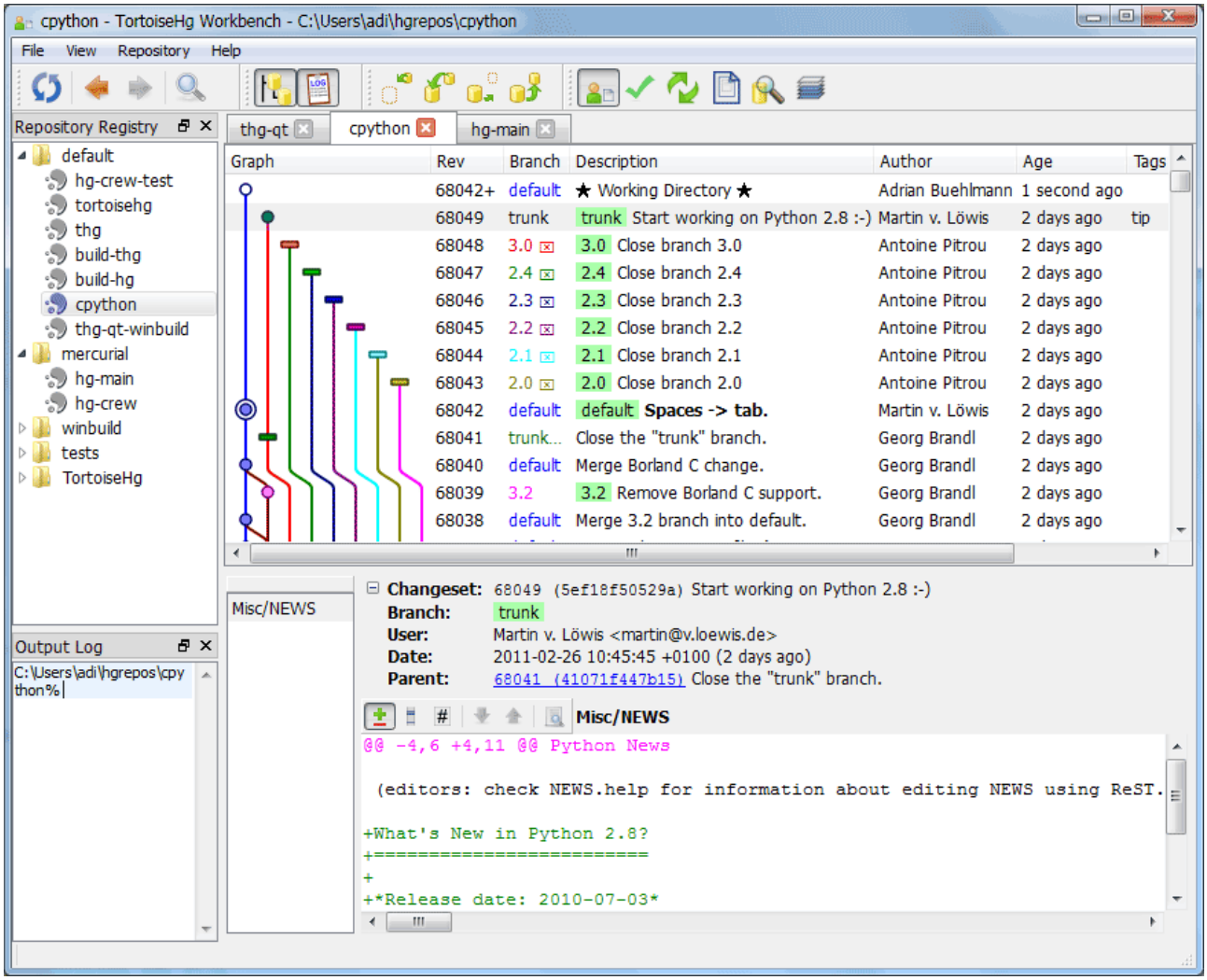
2.2 Opis delovne prakse – dela študenta

Delovno prakso sem opravljal kot programer in ker v podjetju nisem bil ravno nov sem za prakso dobil projekt Slovar. Ampak ker je bil moj računalnik ravno sveže naložen sem moral za začetek računalnik dati v domeno od podjetja, da sem lahko imel dostop do vseh podatkov, strežnikov,…. Nato je bilo potrebno naložiti vsa potrebna orodja za delo. Za začetek sem moral dobiti vso programsko kodo ki jo potrebujem. Za programsko kodo smo uporabljali programsko orodje CVS, ki je namenjen shranjevanje vseh podverzij, tako da je nemogoče narediti veliko škode podjetju, če je ta napaka hitro zaznana. Ker je bil CVS star se je podjetje odločilo da bo zamenjalo in menjava se je dogajala ravno med mojo delovno prakso. Odločitev je padla na distribucijsko orodje Mercurial, ampak to se je zgodilo blizu polovice moje delovne prakse.



Slika 3: Staro orodje za shranjevanje podverzij, CVS.

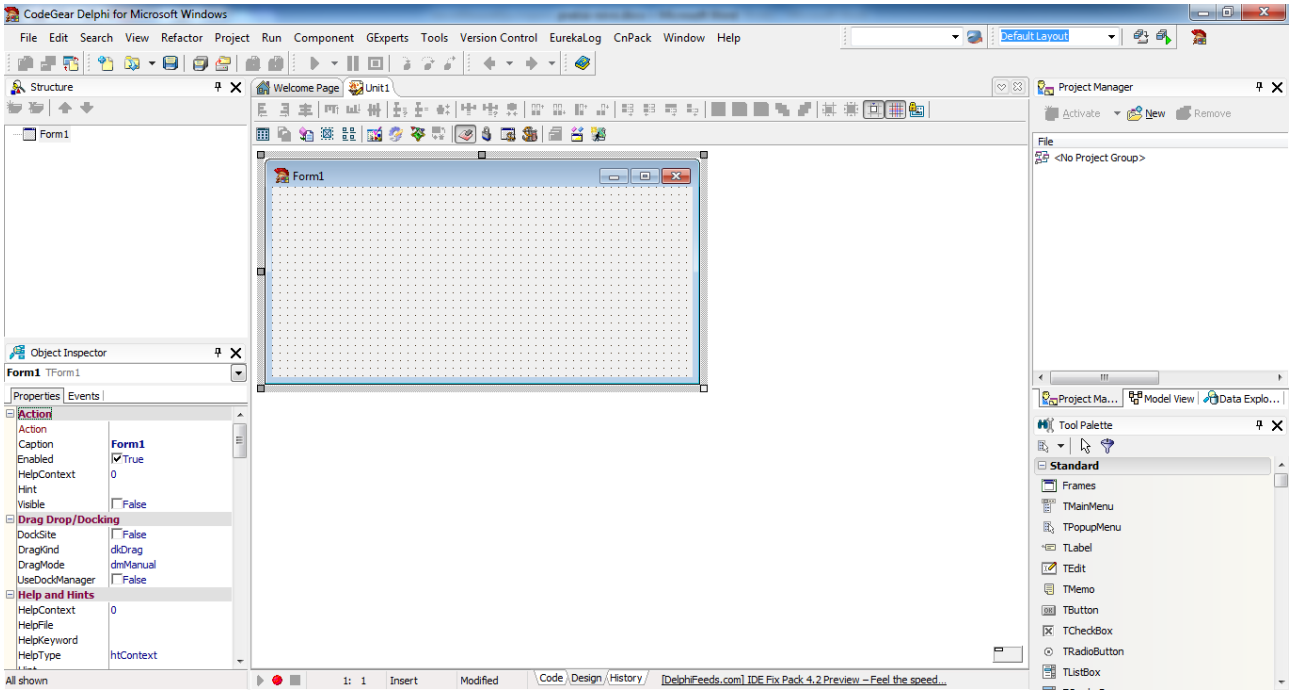
Za začetek je potrebno dobiti vso kodo iz CVS strežnika, tako da jo imamo lokalno na računalniku. Vsaka sprememba ki je narejena v tej kodi jo winCVS obarva rdeče kar pomeni da je spremenjena. Kodo lahko primerjamo s trenutno na strežniku da vidimo narejene spremembe in če nam je vse všeč jo lahko naložimo na strežnik. V tem primeru nam CVS shrani naše uporabniško ime, spremembe, ki smo jih naredili, datum, ter naš komentar.



Slika 4: Novo orodje za shranjevanje podverzij Mercurial.

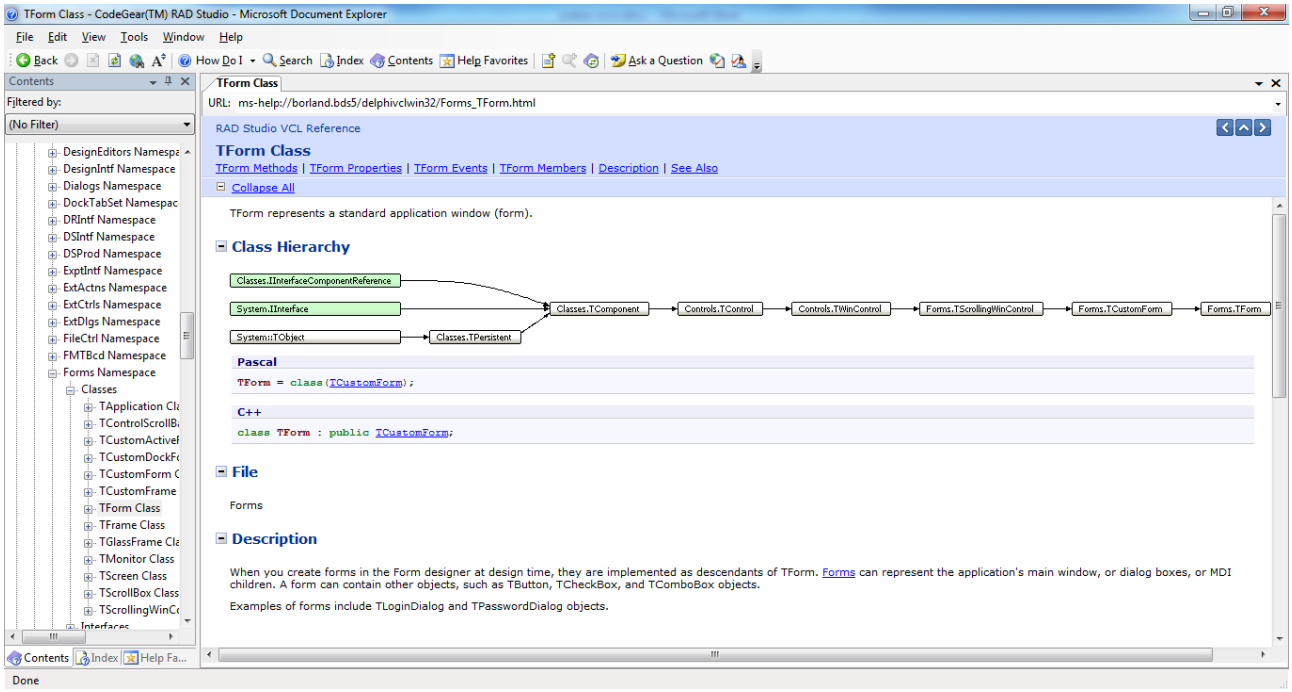
Po menjavi se je postopek za vzdrževanje kar precej spremenilo. Ko dobimo vso kodo iz Mercurial strežnika se nam lokalno naredi strežnik ki je klon tiste na originalnem strežniku. Nato mi lahko lokalno spreminjamo, dodajamo na lokalni strežnik kot smo na CVSju delali direktno na strežniku. Prednost je bila da smo lahko delali po koščkih in ni bilo vse shranjeno v neko celoto. Ko je bila naša naloga opravljena smo originalnemu strežniku poslali vse spremembe, ki so nastale na lokalnem strežniku. Originalni strežnik je preveril če so bile že narejene kakšne spremembe od drugih zaposlenih pred tem in če težav ni bilo je vse spremembe vzel ter jih shranil.

Ko sem imel vso programsko kodo, sem si moral tudi naložiti vsa delovna orodja. Potreboval sem Microsoft Office 2010 za branje specifikacij ter pisanje dokumentacije, Microsoft SQL Server 2008 za dostop do SQL baze, ter zame najpomembnejše CodeGear Delphi 2007 ter vse komponente za delovanje pixi* aplikacij.



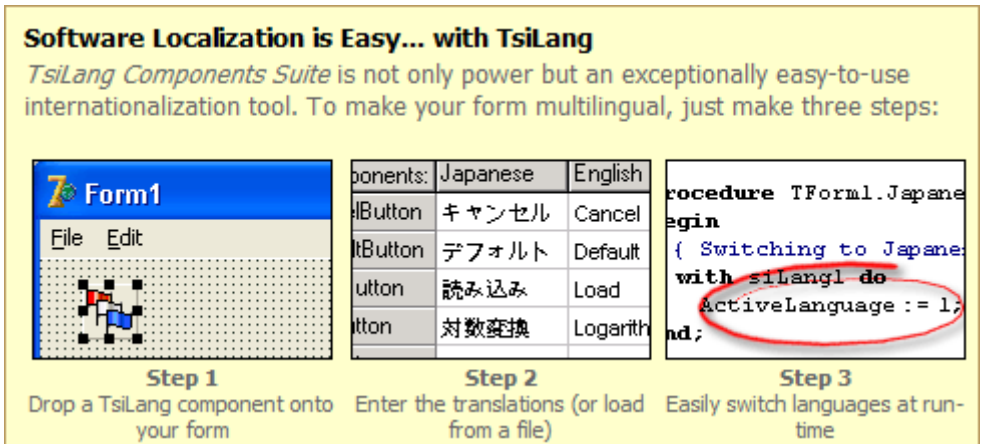
Slika 5: Programsko okolje CodeGear Delphi 2007.

Najtežji oziroma najbolj dolgotrajni del je bil naložiti in nastaviti vse CodeGear Delphi 2007 komponente, saj jih je bilo zelo veliko. CodeGear Delphi 2007 uporablja jezik Delphi, ki je baziran na jeziku Pascal. Pascal je že star a vsekakor ne neuporaben, Delphi pa se še razvija naprej. Programsko okolje CodeGear Delphi 2007 je kar pregledno, zelo me pa motu ker nima lepo napisane dokumentacije, kar se mi zdi zelo pomembno pri učenju jezika oziroma iskanju rešitve.



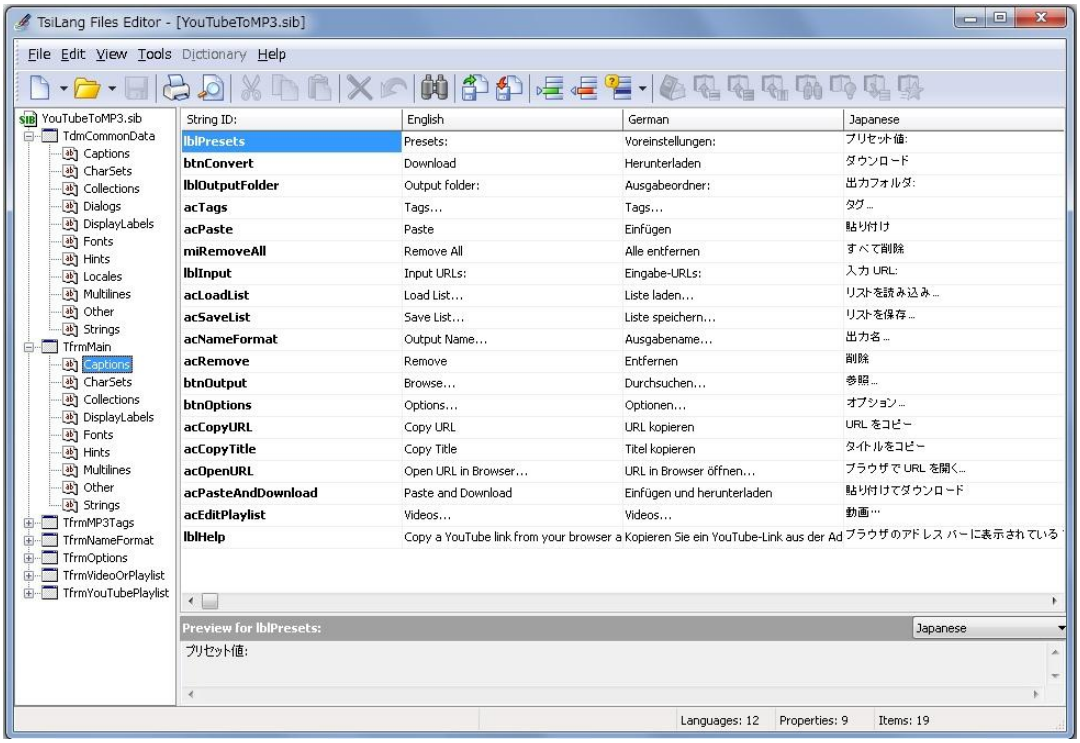
Slika 6: Dokumentacija za programsko okolje CodeGear Delphi 2007.

Ker sistem pixi* ni namenjen le eni državi to pomeni da tudi potrebuje več jezikov. Prvotno je bilo to opravljeno kar direktno v aplikaciji prek za to namenjene komponente. Ker je bilo popravljanje in dodajanje prevodov zamudno delo in povrh še nezanimivo se je podjetje odločilo da se naredi vse skupaj ločeno od aplikacije. Ker delovnega orodja in komponente nismo želeli menjati sem začel raziskovati primerne rešitve. Ta komponenta se je imenovala TsiLang.



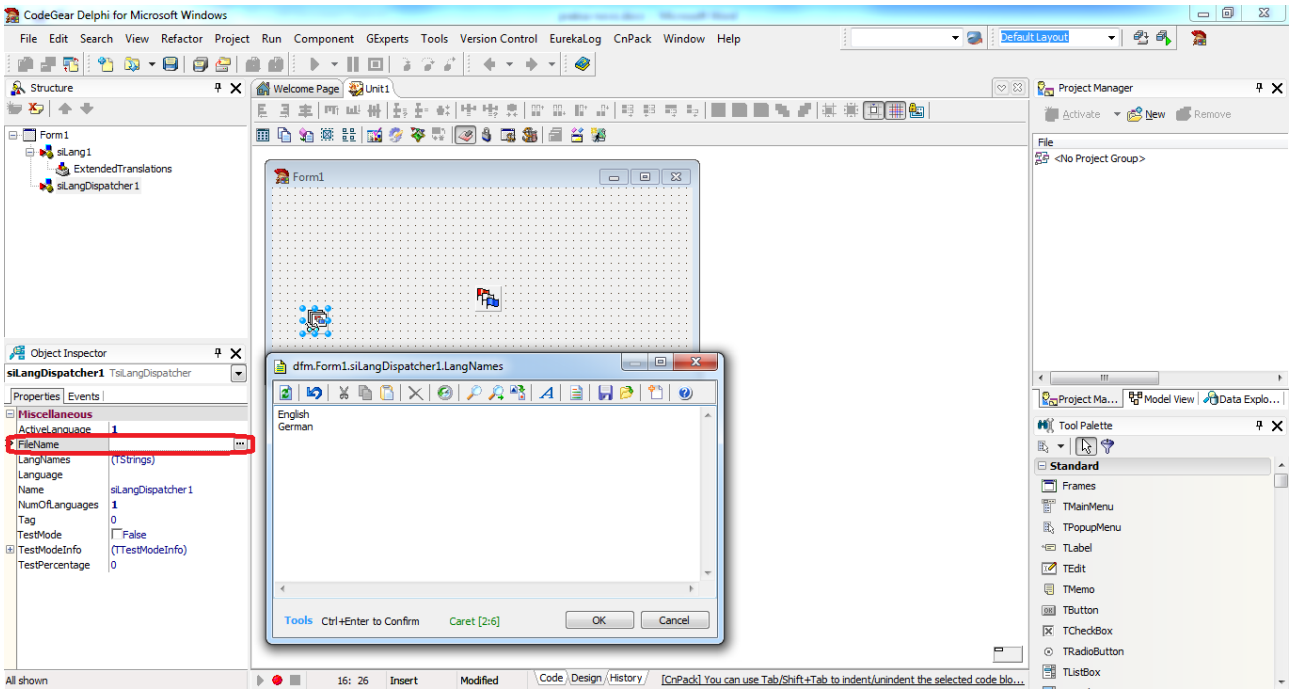
Slika 7: Komponenta TsiLang in hiter način uporabe.

Po nekaj raziskovalnih dneh sem našel, da TsiLang podpira tudi branje in ne le shranjevanje v zunanje datoteke. Shranjevanje je bilo mogoče prek njihovega programa SilEditor.



Slika 8: Program SilEditor.

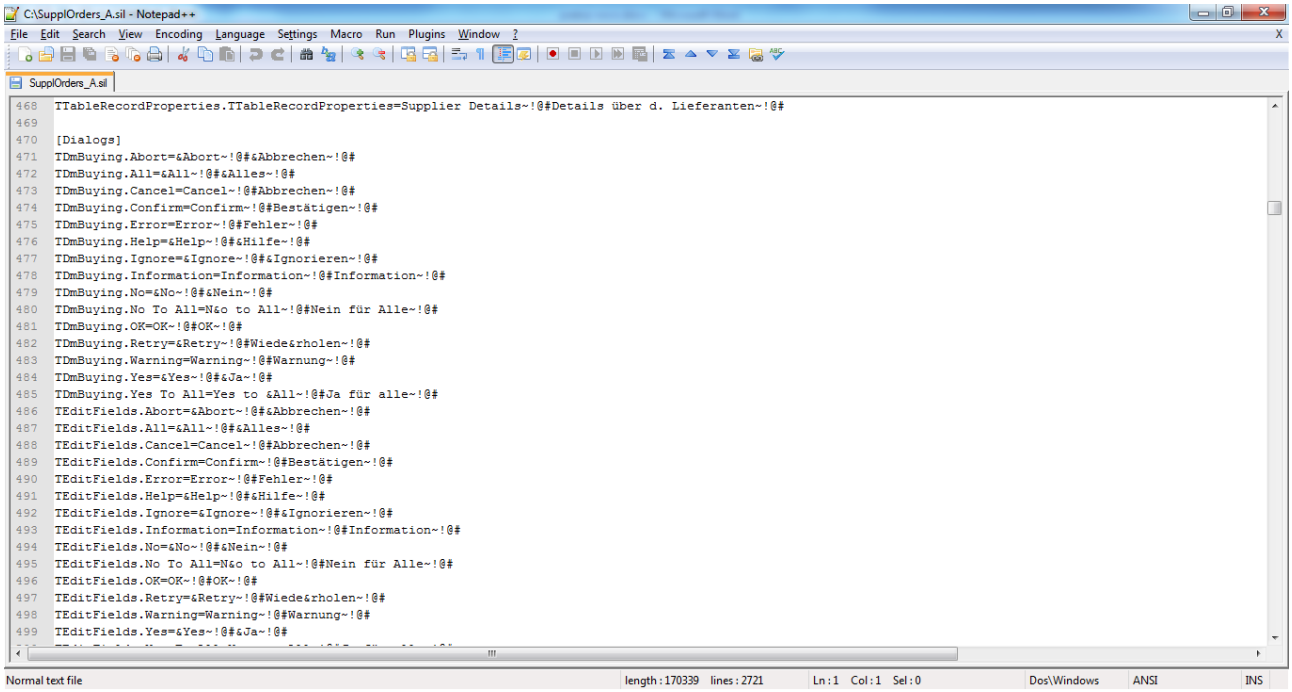
Preko SilEditor sem nato urejal ter dodal vse prevode iz aplikacij ter jih shranil v enako ime kot ga ima aplikacija. V aplikacije sem datoteko prebral preko siLangDispatcher komponente, katere avtor je tudi TsiLang.



Slika 9: Komponenta siLangDispatcher ter njene lastnosti.

Ker se veliko besed ponavlja je bila ideja tudi, da se išče po besedah in ne po komponentah programa, ampak ker so bile besede dostikrat drugačne v nekaterih jezikih, v angleščini pa isti se to ni implementiralo. S tem bi lahko prevajalcu zelo olajšali delo, ter bi bilo tudi najbolje optimizirano. Vse bi se sklicevalo na en veliki slovar, ter ne bi bilo potrebno za vsako aplikacijo posebej se sklicevati na svojega. Nato je postal še problem hitrosti. Velike aplikacije so seveda vsebovale veliko besed in s tem posledično več časa za branje te zunanje datoteke. Vse besede so se naložile na začetku, zato da je aplikacija nato delala hitro. Ampak tudi na začetku se večino uporabnikom ne da čakati po par sekund, mogoče celo deset ali več. Na žalost se je izkazalo da ni deset sekund ampak veliko več, minuta. Ta rešitev ni bila več sprejemljiva in začel sem iskati nove optimizacijske rešitve. Ta problem sem pa rešil s pretvarjanjem iz tekstovnega formata v binarnega. To sem ponovil na vseh aplikacijah in s tem je bila moja delovna praksa končana.

Sedaj se uporabljata obe datoteki, tekstovna in binarna. Tekstovna je zelo uporabna za pregled na Mercurial, saj je zelo lahko najti napako oziroma spremembo. Sintaksa tekstovne datoteke, ki ima končnico .sil, je zelo enostavna tudi brez programa SilEditor. Zelo podobno strukturo ima kakor .ini datoteke.



Slika 10: Tekstovna datoteka za prevajanje .sil.

Če bi v Mercurial dodajali binarne datoteke bi se pojavil problem, ker noben ne bi vedel kakšne spremembe so se zgodile. Binarne datoteke imajo končnico .sib. Končna odločitev je nato bila da se uporablja počasno tekstovno datoteko med razvojem oziroma če prevodov ne potrebujemo se jih sploh ne vključuje v projekt, zato da pohitrimo. Ko pa gredo prevodi k strankam pa se pretvorijo v binarno obliko ter pošljemo njim. V podjetju se uporabljajo samo za testiranje prevodov.

3. SKLEP

V tem podjetju sem delal že prej ampak nikoli tako obsežno kot med to delovno prakso, tako da se mi zdi delovna praksa zelo uporabna. V primeru da še ne bi nikjer imel dela bi bil to odličen začetek saj v štirih tednih z solidnim znanjem programiranja lahko narediš veliko. Ko sem dobil nov projekt mi je šlo vse skupaj bolj počasi na začetku, ampak ko sem našel rešitev ter začel to implementirati je vse skupaj postalo veliko lažje. Projekta še nisem dokončno dokončal, ampak v večini primerov je ideja, ki je izvedljiva vredna veliko več. Seveda ga imam namen tudi dokončati. Delovna praska je potekala od ponedeljka do petka, začetek je bil poljuben od sedmih do devetih in seveda osem ur odkar sem prišel. Po parih urah dela smo se vsi zaposleni in študenti zbrali v sejni sobi ter na kratko opisali naloge in probleme ki smo jih imeli v preteklem dnevu in kaj nameravamo delati danes. Za komunikacijo z sodelavci iz Nemčije smo uporabljali programsko orodje Skype. Med opravljanjem delovne praske smo tudi menjali orodje za shranjevanje programske kode (subversion system). Odšli smo iz CVS na Mercurial. Ta preskok se mi je tudi zdel kar težak saj sem bil (kot tudi vsi ostali) navajen starega orodja, vendar smo se po kakem tednu lepo navadili. Poleg opisanega dela zgoraj je to vse kar sem doživel v teh štirih tednih, kar seveda ni ravno malo.

4. LITERATURA IN VIRI

[1] (2011) Uradna slovenska stran za geniusnet d.o.o.. Dostopno na: http://www.geniusnet.si/.

[2] (2011) Uradna nemška stran za pixi*. Dostopno na: http://www.pixi.eu/.

[3] (2011) Interni portal za sistem pixi*. Dostopno na: https://portal.madgeniuses.net:550/.

[4] (2011) Uradna spletna stran za TsiLang. Dostopno na: http://www.sicomponents.com/.

[5] (2011) Spletna stran za CVS. Dostopno na: http://cvsgui.sourceforge.net/.

[6] (2011) Spletna stran za Mercurial. Dostopno na: http://tortoisehg.bitbucket.org/.