Jedna implementacija ETL procesa upotrebom Talend Open Studija

Emir Pećanin
Departman za matematičke nauke
Državni univerzitet u Novom Pazaru
Novi Pazar, Srbija
epecanin@gmail.com

Siniša S. Ilić
Katedra za računarstvo i informatiku
Fakultet tehničkih nauka Kosovska Mitrovica
Kosovska Mitrovica, Srbija
sinisa.ilic@pr.ac.rs

Petar Spalević
Katedra za elektroniku i telekomunikacije
Fakultet tehničkih nauka Kosovska Mitrovica
Kosovska Mitrovica, Srbija
petarspalevic@yahoo.com

Sažetak - **U ovom radu smo predstavili jedan jednostavan i efikasan način transformacije podataka koristeći Open Studio kompanije Talend. Opisali smo ETL proces od trenutka povezivanja softvera i transakcione (OLTP) baze pa do učitavanja transformisanih podataka u skladište podataka (OLAP). Na kraju smo dali smernice kako se ovaj pristup može koristiti u automatizaciji celog procesa transformacije podataka.**

Ključne reči – ETL; Skladište podataka;

I. UVOD

Razvoj i implementacija informacionih sistema u preduzećima, velikog i malog obima, doprinele su povećanju efikasnosti čuvanja i obrade podataka. Danas se gotovo ne može zamisliti poslovanje a da u pozadini istog ne postoji sistem koji će svakodnevno evidentirati poslovne informacije datog preduzeća i smestiti ih u odgovarajućoj, transakcionoj bazi.

Baze su organizovane na takav način da mogu prihvatiti unos i modifikaciju velike količine informacija a da se pritom mora poštovati princip normalizacije (ne-redundantnosti) podataka u tabelama. Informacije o individualnim transakcijama, iako bitne preduzećima, nisu toliko bitne analitičarima podataka jer iz njih, analizom, ne mogu se izvući potrebni zaključci koji bi pospešili poslovanje preduzeća [1].

Baze se uređuju i čiste od nepravilnih i nepotrebnih informacija. Tako transformisani podaci se, na kraju procesa obrade, smeštaju u skladište podataka (Data Warehouse).

Skladište podataka je nezavisna baza podataka u smislu da je odvojena od transakcione baze preduzeća. Skladište pruža pregled generalizovanih i ujedinjenih podataka putem višedimenzionih pogleda (multidimensional view) ili putem OLAP (OnLine Analytical Processing) alata. Osim toga, dopušta analitičarima da koriste razne data-mining metode kao što su klasterizacija, klasifikacija i predikcija u cilju poboljšanja procesa analize podataka [2]. Ukratko, skladište predstavlja kopiju transakcione baze podataka, specijalno prilagođene za pisanje upita i analizu podataka [3].

Pre nego što se podaci mogu analizirati potrebno je da prođu kroz ETL proces.

ETL proces se sastoji od sledećih faza:

- Preuzimanje podataka (Extract) – relevantni podaci se iz transakcione baze preuzimaju radi obrade,
- Uređivanje podataka (Transform) - podaci se čiste od nepotrebnih i nepravilno unetih informacija, spajaju ili modifikuju radi lakše analize,
- Učitavanje podataka (Load) – uređeni podaci se unose u skladište podataka.

ETL je samo jedna od faza organizacije skladišta podataka, na koju se utroši i do 70% vremena posvećenog poslu [4].

U ovom radu predstavićemo kompletan ETL proces koristeći Talend-ov open source programski paket Open Studio for Data Integration.

II. TALEND OPEN STUDIO FOR DATA INTEGRATION

Proces transformacije podataka zahtevao je izradu softvera koji bi mogao da podmiri zahteve korisnika i izvrši potrebne transformacije nad podacima a da ujedno bude relativno lak za korišćenje. Danas, postoji veliki broj komercijalnih softverskih rešenja koje preduzeća koriste u procesu tranformacije podataka. Mi smo se, u ovom radu, odlučili za open source programski paket kompanije Talend, i to Open Studio for Data Integration.

Open Studio je razvojno okruženje sa grafičkim interfejsom koji nudi veliki broj alata za obradu i integraciju obrađenih podataka kako malim tako i velikim preduzećima. Open Studio se oslanja na grafičku reprezentaciju procesa transformacije koristeći prikaz koji najviše podseća na dijagram toka koji povezuje procese onim redosledom kojim će se vršiti transformacija podataka.

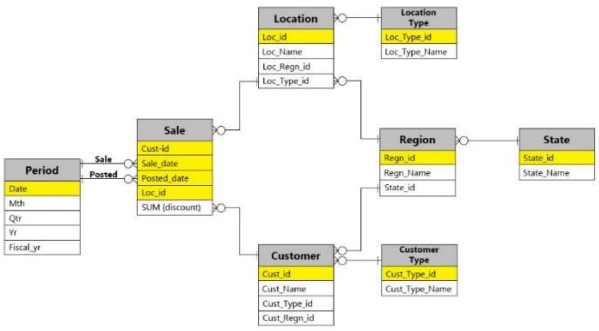
Studio je pisan u Javi pa pored SQL upita, u zavisnosti od toga koliko dobro poznaje Java kod, korisnik može i programirati određene komponente koje Talend nije još uvek uvrstio u biblioteku postojećih.

Terminologija Talendovog Open Studija se razlikuje, u određenoj meri, od ostalih softverskih rešenja koja se bave ETL-om. U Studiju, komponenta ili skup komponenti koji vrše integracioni proces se prosto naziva poslom (job).

III. OPIS BAZE KOJU ĆEMO KORISTITI U NAŠEM PRIMERU

Da bi smo demostrirali mogućnosti Open Studija kreirali smo jednostavnu transakcionu, OLTP, bazu [5] koja je prikazana na slici 1. Baza sadrži informacije o tome ko je kupio robu, kada je kupljena i lokaciji prodaje. Sastoji se od tabele *Period* koja je na slici prikazana na levoj strani u kojoj se nalaze informacije o vremenskom periodu prodaje, mesecu, godini i fiskalnoj godini. Na desnoj strani slike se nalazi tabela *Sale* koja sadrži informacije o prodaji, datumu prodaje i lokaciji i tabele koje sadrže informacije o lokaciji, regionu i o kupcu. Ovu bazu smo nazvali ***projectDB***.

Kako su nam potrebni svi podaci iz početne baze, ne postoje kolone koje je potrebno zanemariti tokom procesa transformacije podataka. Međutim, u praksi, to je redak slučaj pa ćemo u nastavku ukazati i na način kako se takve kolone mogu izostaviti iz procesa transformacije.



Slika 1. Relacioni diagram baze

IV. PRIPREMA PODATAKA ZA ANALIZU

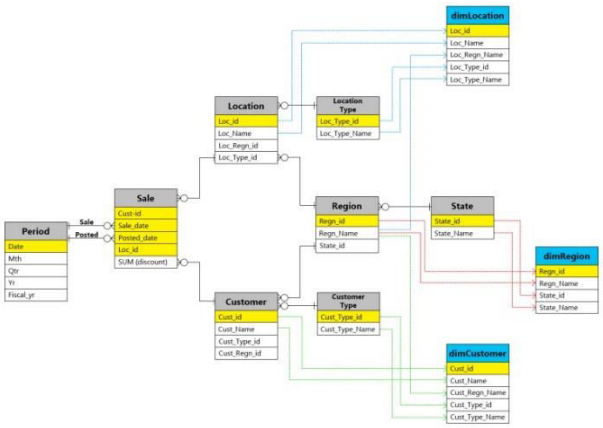
Skladište podataka, prema karakteristikama, je predmetno orjentisano (object – oriented) [6], odnosno, organizuje se u zavisnosti od tipa podataka nad kojima je potrebno izvršiti analizu. Kompleksne transakcione baze se dodatno pripremaju pre ETL procesa. Uklanjaju se strani ključevi kako bi se izbegle dodatne komplikacije. U studiju je ceo proces pripreme pojednostavljen pa, iako koristimo transakcionu tabelu kao osnovu, Studio je neće tretirati kao relacionu bazu. Svaka tabela u bazi će biti zaseban entitet. Informacije o primarnim i stranim ključevima u tabelama se pritom ne gube.

Dimenzione tabele (sl. 2) su organizovane na sledeći način:

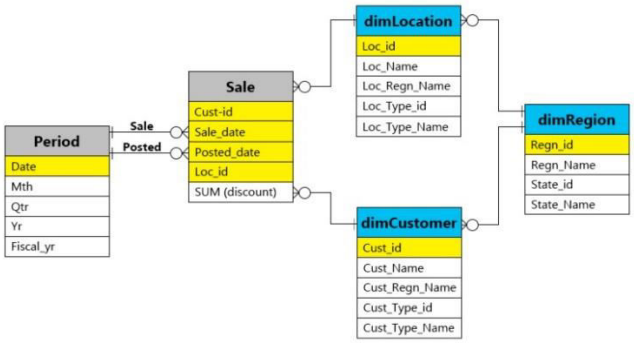
- Tabela **dimLocation** sadrži ime lokacije, tip lokacije i region u kome se nalazi. Tabela će preuzeti informacije o regionu na osnovu stranog ključa tabele Location i uneti imena iz tabele Region.
- Tabela **dimRegion** sadrži primarne ključeve i imena tabele Region i tabele State.
- Tabela **dimCustomer** sadrži ime i primarni ključ tabele Customer, tip i region kome pripada.
- Tabele **Sale** i **Period** smo prosledili skladištu bez transformacije kao factSale i factPeriod.

Na slici 2 tankim (plavim) linijama grafički je pokazano koje su kolone preuzete iz odgovarajućih OLTP tabela kako bi se kreirale dimenzione tabele

Relaciona šema skladišta se može videti na slici 3.



Slika 2. Kreiranje dimenzionih tabela



Slika 3. Relaciona šema skladišta podataka

V. IMPLEMENTACIJA ETL PROCESA UPOTREBOM TALEND OPEN STUDIJA

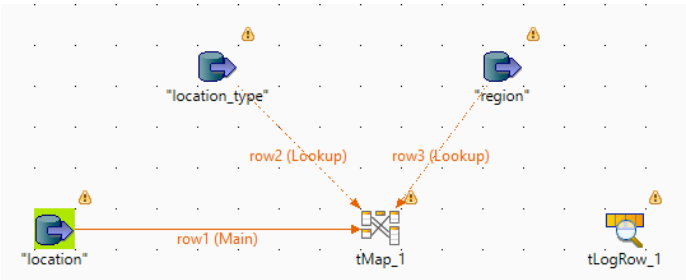
Na samom početku potrebno je uspostaviti vezu između Studija i baze ***projectDB***. Povezivanje je jednostavno i vrši se preko vizarda kome pristupamo preko kontesktnog menija kategorije **Db Connections**. Kada napravimo vezu, u kategoriji Db Connections pojaviće se naziv baze sa kojom smo povezali program. Potrebno je preuzeti šeme kako ih bismo prosledili komponentama Studija koje će te iste podatke transformisati.

Tabele i kolone u novoj šemi će zadržati podešavanja iz starih, međutim, Studio nam pruža mogućnost modifikacije određenih parametara šeme (tip, ime kolone, preciznost itd.). Koristeći obimnu biblioteku komponenti kreirali smo posao kojim ćemo obaviti ETL proces. Osim komponenti iz biblioteke i tabele koje smo preuzeli iz transakcione baze, možemo koristiti u obliku komponenti unutar programa.

Komponente koje smo koristili prilikom definicije posla:

- *tMySqlInput* - komponenta za unos (čitanje) podataka iz MySQL baze;
- *tMySqlOutput* – komponenta kojom ćemo učitati obrađene podatke u skladište.
- *tMap* – komponenta kojom definišemo pravila i način spajanja tabela naše baze podataka. Poseduje sopstveni Java editor u kome možemo precizno definisati izmene nad podacima unutar baze;
- *tLogRow* – komponenta koja prikazuje rezultat izvršavanja posla na standardnom izlazu studija. Koristimo je kako proverili validnost rezultata. Na kraju provere ćemo je zameniti *tMySqlOutput* komponentom.

Studio pravi razliku među povezanim komponentama prema redosledu povezivanja istih pa se prva input komponenta, koju povežemo sa tMap komponentom, tretira kao glavna tabela dok će svaka naredna biti lookup tabela (sl. 4).

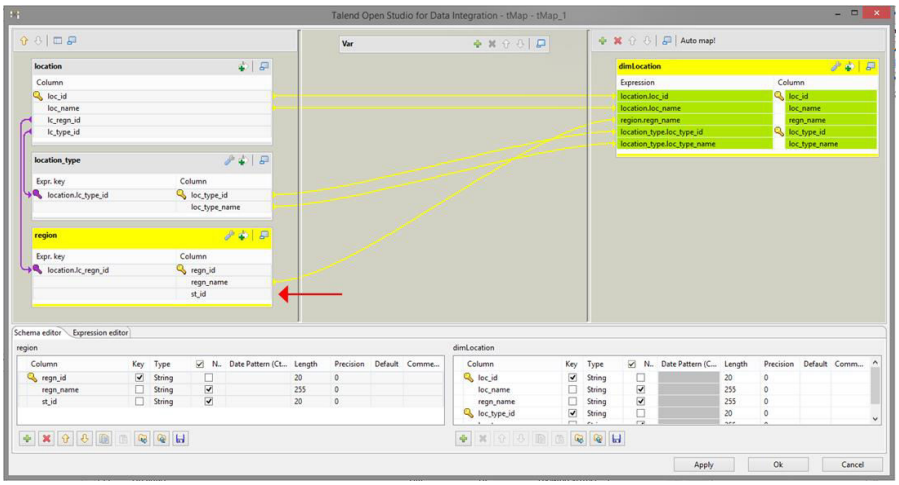


Slika 4. Dizajn dimLocation tabele

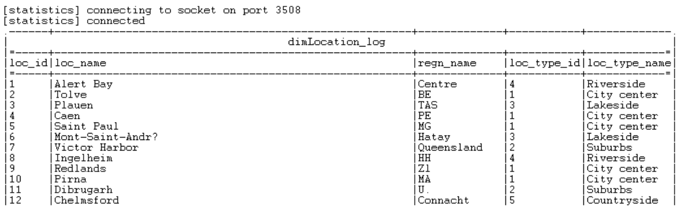
Izvršili smo agregaciju podataka, iz tabela **Location**, **Location_type** i **Region,** pomoću tMap komponente tako što smo kao uslov spajanja izabrali *Left Outer Join*.

Povezivanje kolona na levoj i na desnoj strani tMap komponente se vrši drag – and – drop tehnikom. Studio će žutim strelicama označiti proces prenošenja podataka. Na osnovu konfiguracije tabela, Studio generiše šemu tabele koja je prikazana na dnu prozora komponente. Kolonu, **st_id** u tabeli **Region**, nismo povezali sa tabelom dimLocation pa ona neće biti uključena u proces transformacije. Pomoću komponente tMap smo definisali primarni ključ tabele dimLocation kao što se može videti na slici 5.

Izlaznu tabelu iz tMap povezali smo sa tLogRow komponentom kako bi proverili rezultat transformacije podataka. Na izlazu (sl. 6.) se može videti da smo uspešno povezali tabele i da smo kao rezultat dobili tabelu koja odgovara tabeli dimLocation.



Slika 5. Konfiguracija tMap komponente



[statistics] connecting to socket on port 3508
[statistics] connected

dimLocation_log

loc_id	loc_name	regn_name	loc_type_id	loc_type_name
1	Alert Bay	Centre	4	Riverside
2	Tolve	BE	1	City center
3	Plauen	TAS	3	Lakeside
4	Caen	PE	1	City center
5	Saint Paul	MG	1	City center
6	Mont-Saint-Andr?	Hatay	3	Lakeside
7	Victor Harbor	Queensland	2	Suburbs
8	Ingelheim	HH	4	Riverside
9	Redlands	Z1	1	City center
10	Pirna	MA	1	City center
11	Dibrugarh	U.	2	Suburbs
12	Chelmsford	Connacht	5	Countryside

Slika 6. Prikaz rezultata

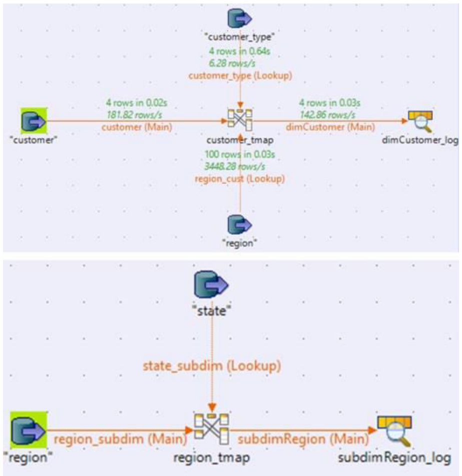
Analogno ovom procesu, konfigurisali smo ETL proces za tabele *dimCustomer* i *subdimRegion* (sl. 7).

Kako je konfiguracija procesa transformacije za sve tabele završena, sada ih možemo međusobno povezati.

Povezivanje grupe procesa, se razlikuje od standardnog povezivanja komponenti prilikom kreiranja posla. Povezivanjem komponenti označavamo tok kretanja informacija tokom izvršavanja posla. Povezivanje grupe procesa je uslovno povezivanje i vrši na osnovu toga da li je grupa procesa uspešno završena ili ne.

Poznavaoci Java programskog koda mogu koristiti tJava komponente kako bi obavestili korisnika porukom da su određeni koraci u procesu završeni. Ili da, ukoliko se javi greška prilikom transformacije podataka prekine izvršavanje i spreči unos netačnih ili pogrešnih informacija u skladište podataka.

Na kraju smo zamenili komponente *tLogRow* sa *tMySqlOutput* komponentama kako bi smo omogućili unos podataka u skladište.

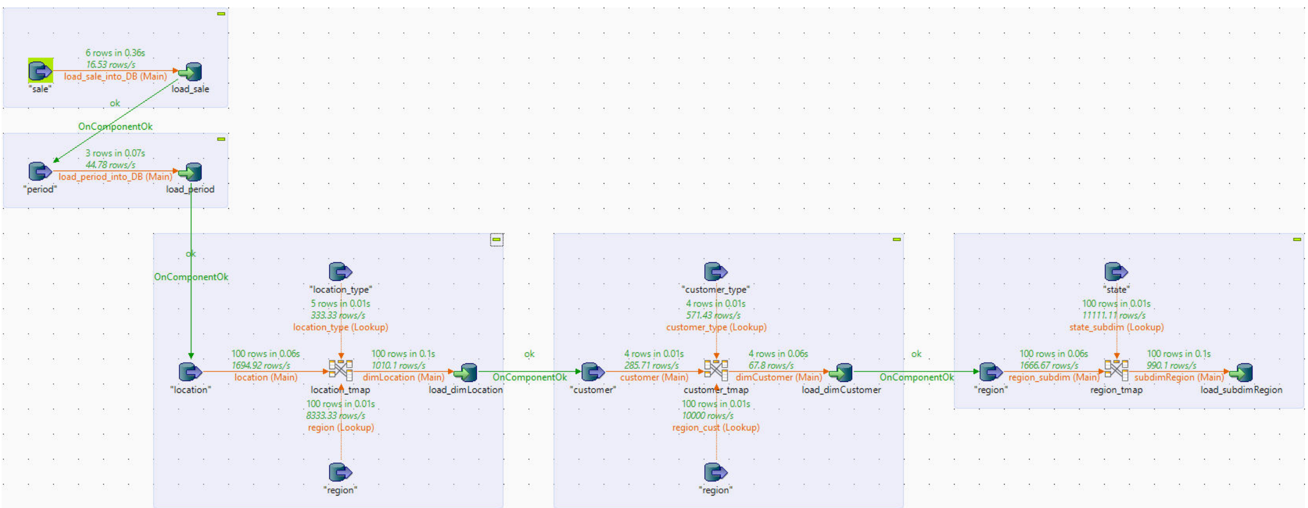


Slika 7. Konfiguracija tabela dimCustomer i subdimRegion

Napravili smo još jednu vezu studija sa ***warehouseDB*** bazom koja je inicijalno prazna. Sve informacije o povezanim bazama Studio čuva u sopstvenom rezpozitorijumu i možemo ih koristiti kako bi uprostili i ubrzali proces kreiranja poslova.

Preostaje nam da definišemo šta će tačno Studio raditi sa podacima. Podešavanja se sastoje od polja "Action on table" i "Action on data". Kako je tabela *warehouseDB* prazna, izabrali smo *Create table if not exist* a za podatke *Insert or update* opciju. Ovo će osigurati da, ukoliko izbrišemo tabelu, Studio napravi novu a ukoliko već postoji tabela da podatke ažurira u istoj.

Pošto su tabele Sale i Period ostale neizmenjene, njih smo prosledili bazi warehouseDB. Konačna šema ETL-a je prikazana na slici 8.



Slika 8. Konačna šema ETL-a

U završnoj fazi imamo izbor da posao ostane kao deo projekta u kome je napravljen ili da ovako uređen pretvorimo u izvršnu datoteku i izvršimo je na računaru na kome Talend nije instaliran..

Kao deo projekta, podaci o tome kako su poslovi kreirani, korisni su nam ukoliko se javi potreba da modifikujemo komponente, da promenimo delove procesa ili kompletan proces. Projekti su organizovani kao zasebne fascikle na hard disku na specijalnoj lokaciji koju definišemo prilikom pokretanja studija.

Kao izvršna datoteka, poslovi postaju nezavisne aplikacije i mogu se pokretati iako nemamo instaliran Studio na hard disku. Aplikacija nema grafički interfejs i informacije o procesu ETL-a prikazuje u konzoli (Dos), ukoliko smo ih definisali prilikom kreiranja posla.

Kreiranje izvršne datoteke je praktično jer dopušta korisnicima da izvrše ETL proces nad bazom bez prethodnog iskustva u radu sa Studiom.

Kreirali smo izvršnu datoteku tako što smo iz kontekstnog menija, koji prikazujemo desnim klikom na ime posla izaberali opciju Export job. Kao tip datoteke je potrebno izabrati *Autonomous Job* kako bi Studio organizovao i "upakovao" sve potrebne Java klasne datoteke i Java biblioteke i napravio samostalnu izvršnu datoteku.

Datoteka se na hard disku skladišti u obliku zip arhive u kojoj se čuvaju izvršne datoteke za operativne sisteme Windows i Linux.

VI. ZAKLJUČAK

U našem radu predstavili smo proces transformacije jedne transakcione baze u skladište podataka (OLAP bazu). Podaci prolaze kroz niz transformacija pre nego što se organizuju, urede, i na kraju, učitaju u skladište. Pokazali smo da Talend Open Studio ima sve potrebne komponente za uspešnu realizaciju ETL procesa jer je dobro struktuirano skladište glavni preduslov da se analiza podataka obavi bez velikih poteškoća. Dodatno, Talend omogućava kreiranje izvršne (exe) datoteke kojom se utvrđeni ETL proces može izvršiti i na sistemu gde Talend Open Studio nije instaliran.

Veliki napor se ulaže da se planiranje ETL procesa skrati i organizacije skladišta olakša. Teži se tome da, kada se faza planiranja završi, proces transformacije baze i učitavanje u skladište svede na jedan klik miša. U ovom radu smo prikazali jednostavan i elegantan način organizacije skladišta podataka i, kreirajući izvršnu datoteku, automatizaciju kompletnog ETL procesa.

Plan budućeg istraživanja je da uporedimo performanse Talendovog Open Studija i drugih, sličnih, komercijalnih i open source, softverskih rešenja.

ZAHVALNICA

Rad je delimično finansiran od strane Ministarstva prosvete i nauke Republike Srbije, u okviru projekta TR35026.

LITERATURA

[1] Siniša Ilić et al., "The Implementation of ETL Processes on example of authorised DLS" in *International Scientific Conference - UNITECH*, Gabrovo, 2015, p. 6.

[2] Liliya Dimitrova and Aleksandar Dimov, "Creation of ETL processes" in *International Scientific Conference - UNITECH*, Gabrovo, 2015, p. 6.

[3] Ralph Kimball. http://www.dwinfocenter.org/defined.html.

[4] Ralph Kimball, Mary Ross, Bob Becker, Joy Mundy, and Warren Thornthwaite, *The Kimball Group Reader: Relentlessly Practical Tools for Data Warehousing and Business Intelligence*. Indianapolis: Wiley Publishing, Inc., 2010.

[5] João Moura Pires, From OLT to OLAP, 2014, Presentation.

[6] Data Warehousing tutorial. (2014) www.tutorialspoint.com. [Online]. http://www.tutorialspoint.com/dwh/dwh_tutorial.pdf

ABSTRACT

Complexity of the ETL process is directly proportional to the size and complexity of the data warehouse we are required to make. Simplification of the technical side of ETL process, allows us to spend more time on planning phase and ensuring that the data, in our data warehouse, is well organized and structured. In this article we have described simple, yet efficient, ETL method using Talend Open Studio for Data Integration. We have described all phases required for the completion of this process. In the end, we have created an executable file, a Studio independent application, containing a complete ETL process. Application can be used with minimal input, allowing user to automatize the ETL process to a certain degree.

ONE IMPLEMENTATION OF THE ETL PROCESS USING TALEND OPEN STUDIO

Emir Pecanin, Sinisa S. Ilic, Petar Spalevic