Vysoká škola ekonomická v Praze
Fakulta informatiky a statistiky
Vyšší odborná škola informačních služeb v Praze

Tomáš Gottwald

Použití SQL Server Integration Services v etapě ETL budování datových skladů

Bakalářská práce 2007

PROHLÁŠENÍ

Prohlašuji, že jsem bakalářskou práci na téma „Použití SQL Server Integration Services v etapě ETL budování datových skladů“ zpracoval samostatně a použil pouze zdrojů, které cituji a uvádím v seznamu použité literatury.

V Praze dne 20. 12. 2006 Podpis

PODĚKOVÁNÍ

Na tomto místě bych rád poděkoval panu Ing. Janu Klasovi z Vysoké školy ekonomické v Praze za vedení mé práce. Mé díky patří rovněž panu Bc. Davidu Kalábovi ze společnosti Adastra, s. r. o., za velmi cennou pomoc při sestavování kapitoly o implementaci DTS na reálném projektu. Dále bych chtěl poděkovat některým dalším kolegům ze stejné společnosti za podnětné rady a připomínky, které mi velmi pomohly při tvorbě práce. Nerad bych opomenul poděkování paní PhDr. Heleně Kučerové za pomoc s formální úpravou dokumentu.

OBSAH

ANOTACE

Tato bakalářská práce se věnuje nástroji Microsoft SQL Server Integration Services (SSIS) a jeho použitím místo Data Transformation Services (DTS) v etapě Extraction Transformation Loading (ETL) budování datových skladů.

Jak známo, ETL je klíčovou a velmi rizikovou fází každého Business Intelligence (BI) řešení. Jedná se totiž o extrakci, transformaci a načítání dat z primárních systémů podniku. A na tom, jaká data budou přenesena do datového skladu, závisí výsledky analýz a reportů, které využívají manažeři k řízení firmy. Celá fáze ETL je velmi komplikovaná a představuje mnohdy majoritní podíl na nákladech celého BI projektu.

Proto vzbudilo uvedení SISS na trh mezi odbornou veřejností velký rozruch. Nejedná se totiž „pouze“ o novou verzi DTS, SISS byly psány od začátku jako úplně nový nástroj. Mají organizaci umožňovat mnohem snadněji integrovat a analyzovat data z heterogenních informačních zdrojů. Nabízejí plnou programovatelnost, integrovatelnost a rozšiřitelnost, což z nich tvoří potenciálně ideální ETL platformu. Jenže vzhledem k obrovským změnám, které nastaly u Microsoftu v ETL etapě, přinášejí s sebou i některá rizika.

Cílem této práce je oba nástroje porovnat a potvrdit či vyvrátit hypotézu, že SSIS je lepším ETL nástrojem než DTS, protože umožňují snadnější integraci dat.

Text je členěn do několika částí, z nichž úvodní se věnuje problematice růstu objemu a významu dat v podnikové sféře. Další část přináší stručný úvod do BI a vysvětluje elementární pojmy a principy, charakteristické pro oblast podnikových informačních systémů. Následuje kapitola pojednávající o ETL. Jsou zde nastíněny typické činnosti a problémy každé fáze procesu. Cílem navazující kapitoly je seznámit čtenáře s nástrojem DTS a především s jeho využitím na reálném projektu. Pak následuje představení SSIS, porovnání jeho možností s DTS a realizace části zmiňovaného projektu pomocí SSIS. Poté prezentuji své závěry a hodnotím SSIS jak na základě znalostí, získaných studiem odborné literatury, tak na základě praktických zkušeností s tímto produktem.

KLÍČOVÁ SLOVA

Business Intelligence, data, data mining, Data Transformation Services, databáze, datový sklad, EAI, ETL, Integration Services, OLAP, OLTP, reporting, SQL, SQL Server.

1 ÚVOD

1. 1 Informační společnost

Žijeme v době, kdy jsou dobré *informace* považovány za nejcennější komoditu. Pro firmy už není problém vyrobit výrobek nebo poskytnout službu. Není dokonce ani velkým problémem výrobek či službu prodat. K tomu ale firma potřebuje mít informace o tom, jaké má zákazníky, které výrobky a služby chtějí a jak je může přimět k jejich koupi.

Historické zkušenosti ukazují, že vývoj celé společnosti do jisté míry kopíruje vývoj v podnikání. Proto dnešní společnost bývá nazývána *informační společností*. Jejími hlavními rysy jsou převaha práce s informacemi, interaktivita, integrace a globalizační tendence. Z technologického pohledu lze říci, že informační společnost je společnost s vysokou mírou využívání *informačních a komunikačních technologií* (*ICT*)[1], založených na prostředcích výpočetní techniky a s nimi spojenou digitalizací [12].

1. 2 Datová exploze

Masivní nasazení ICT (nejen) do podnikové sféry umožňuje uživatelům velmi snadné zpracování, výměnu, sdílení a ukládání dat. Množství zpracovávaných dat přitom neustále exponenciálně narůstá. Zpočátku se data příliš neměnila a docházelo spíše ke změnám samotných programů, které nad těmito daty pracovaly. Výpočetní technika byla využívána spíše k urychlení a automatizaci složitých matematických úloh a nahradila celé skupiny pracovníků a jejich logaritmická pravítka, později kalkulačky. Data byla obsažena v jednom modulu s programovou logikou, která je zpracovávala. Tato data vznikala a zanikala současně s během programu [26].

Postupem času začalo dat přibývat a výpočetní technika zároveň pronikla i do ekonomiky, výroby, administrativy, atd. Vztah mezi daty a programy se začal obracet. Data postupně přibývala a u obslužných programů naopak došlo ke stabilizaci. Vznikla řada aplikací, které pracovaly nad stále bobtnajícími a měnícími se daty. Požadavky na ukládání dat, jejich organizaci a rychlý přístup k nim, se začaly zvyšovat. V rámci programů se objevily speciální souborové struktury, ve kterých se data uchovávala už nikoli pouze při vyvolání programu v paměti, ale přímo na disku počítače. Tento trend byl samozřejmě podmíněn zvyšováním kapacity pevných disků tehdejších počítačů. Na tomto místě je potřeba jmenovat programovací jazyk Cobol, který je vzhledem ke svým datovým strukturám považován za přímého předchůdce *databází*[2] a v němž byla vyvinuta celá řada ekonomických a bankovních aplikací [38].

[1] Rozvoj ICT bývá označován jako příliv tzv. páté Kondratěvovy vlny. Teorie dlouhých Kondratěvových cyklů (nebo vln) modelově člení dvousetletý interval od průmyslové revoluce k revoluci vědeckotechnické do čtyřech cyklů po zhruba padesáti letech: průmyslová revoluce, věk železnic (a lodní dopravy), technicko-vědecká revoluce a (s obráceným důrazem na vědu) vědecko-technická revoluce [46]. Jako pátá vlna navazuje informační společnost.

[2] Databáze je systém sloužící k modelování objektů a vztahů reálného světa (včetně abstraktních nebo fiktivních) prostřednictvím digitálních dat uspořádaných tak, aby se s nimi dalo efektivně manipulovat, tj. rychle vyhledat, načíst do paměti a provádět s nimi potřebné operace – zobrazení, přidání nových nebo aktualizace stávajících údajů, matematické výpočty, uspořádání do pohledů a sestav [23].

1. 3 Databáze

Přístup k datům a jejich organizace byly ale stále poměrně komplikované a programátorsky náročné. To se zlepšilo s příchodem databází, tedy oddělením datových struktur od vlastního obslužného programu. Tento obslužný program nazýváme *Database Management System* (*DBMS*) neboli *Systém řízení báze dat*. První databáze měly *hierarchický model*, který se blíží reálnému uspořádání světa. U tohoto modelu je typická práce se stromy, kde jsou realizovány vztahy 1:N[3]. Variací hierarchického modelu je *síťový model* databáze, který umožňuje vyjadřovat i vztahy N:M[4]. Fyzická realizace síťového modelu je ale náročná a aktualizace obvykle komplikovaná, navíc jak hierarchický, tak síťový model není uzpůsoben pro dotazy [36].

Zmiňované modely překonává *relační model*, který ve své práci publikoval v roce 1969 Dr. Edgar „Ted" F. Codd. Tento model definuje způsob, jakým je možné reprezentovat strukturu dat, způsoby jejich ochrany a operace, které můžeme nad daty provádět. Relační databáze je sestavená z řady tabulek, jejichž sloupce jsou vázány na sloupce v jiných tabulkách. Takto propojená datová pole jsou na sobě určitým způsobem závislá. Jejich vztahy jsou založeny na klíčových hodnotách uložených v příslušných sloupcích. Výhodou relačních databází je jejich relativně snadná modifikace a propojování tabulek [36]. Práce Dr. Codda se stala základem výzkumného projektu System/R, který probíhal v sedmdesátých letech v laboratořích IBM.

K používání modelu byl v IBM vyvinut *SEQUEL* (*Structured English Query Language*)[5], který byl později přejmenován na *SQL*. V roce 1979 firma Relational Software, Inc. (dnešní Oracle) představila první komerční implementaci SQL [48]. SQL se stal nejrozšířenějším dotazovacím (neprocedurálním[6]) programovacím jazykem určeným k definici, údržbě a vyhledávání dat v relačních databázích. Pro manipulaci s jednotlivými záznamy se využívá množina dále nedělitelných transakčních operací, jako je například přidání záznamu, změna již existujícího záznamu a jeho odstranění. Kolekce více tabulek, jejich relací, indexů (řazení podle vybraných sloupců) a dalších součástí tvoří relační databázi [38].

1. 4 OLTP

Relační databáze, využívané v dnešních podnicích, bývají označovány jako *transakční*. Hlavním požadavkem na tyto uložiště dat je, aby v co nejkratším čase dokázaly zpracovat velké množství transakcí[7]. Výsledek transakce je buď *commit* (promítnutí změn do databáze) nebo *rollback* (návrat do původního stavu, resp. neprovedení změn naakumulovaných v průběhu transakce [33]. Další charakteristikou transakčních systémů je možnost víceuživatelského přístupu, což znamená, že k jednotlivým záznamům tabulek může v jednom okamžiku přistupovat více uživatelů, kteří mohou tyto záznamy změnit. Mechanismy zamykání a uvolňování záznamů a s tím související datové bezpečnosti se dříve musely poměrně pracně programovat, kdežto dnes jsou součástí vlastní logiky databáze [38]. Pro zefektivnění práce s tabulkami se návrh relační databáze upravuje podle tzv. *normálních*

[3] Nejjednodušším vztahem je vztah 1:1 (one-to-one), což znamená, že první entitě (záznamu v databázové tabulce) odpovídá maximálně jedna jiná entita (záznam v jiné tabulce). Relaci lze zajistit pomocí unikátních klíčů v obou dvou tabulkách. Např. zákazník uvedl právě jedno číslo účtu a účet patří jen jednomu zákazníkovi. Vztah typu 1:N (one-to-many) znamená, že jednomu záznamu v jedné tabulce může odpovídat jeden nebo více záznamů v jiné tabulce. Např. k zákazníkovi se může pojit více objednávek, ale objednávka je vystavena jednomu zákazníkovi.

[4] N:M (many-to-many), znamená, že více záznamů z jedné tabulky může být svázáno s více řádky v jiné tabulce. Např. v jedné objednávce může být více druhů zboží a jeden druh zboží může být uveden ve více objednávkách. Vzhledem k tomu, že většina současných databázových systémů nedokáže pracovat přímo se vztahy N:M, používá se tzv. vazební entita, což je vlastně spojovací tabulka, která rozdělí vztah N:M na dva vztahy typu 1:N.

[5] SEQEL měl totiž co nejvíce napodobovat běžný jazyk (angličtinu) [36].

[6] To znamená, že zadáváme „jakou" operaci chceme vykonat, nikoli „jakým způsobem" se má vykonat.

[7] Transakce je logická jednotka zpracování dat, která se skládá z jednoho nebo více SQL příkazů provedených jedním uživatelem [31].

forem[8]. Cílem je zejména odstranění *redundance*[9] v datech. I když platí, že čím vyšší normální forma, tím lépe by se s tabulkami z hlediska aplikační logiky mělo pracovat, obvykle se v praxi používá třetí normální forma (3NF). Boyce/Coddova, čtvrtá a pátá normální forma jsou určeny pro speciální případy [37].

V případě, že relační databázový systém pokrývá většinu podnikových aktivit, nazýváme ho systém *ERP* (*Enterprise Resource Planning*) [25]. Data jsou zde přehledně uspořádána a pokud je správně navržena datová základna, jednotlivé transakce se provádějí rychle a zadané dotazy mají adekvátní dobu odezvy. Navíc zajišťují integritu dat, bezpečnost přístupu k datům a další potřebné charakteristiky spojené s řízením firmy na taktické nebo operační úrovni [29].

[8] Normální formy: Nultá (0NF) – existuje-li alespoň jedno pole, obsahující více než jednu hodnotu, První (1NF) – do každého pole lze dosadit pouze jednoduchý datový typ (jsou dále nedělitelné), Druhá (2NF) – jestliže je v první a navíc platí, že existuje klíč a všechna neklíčová pole jsou funkcí celého klíče (a tedy ne jen jeho částí) [41], Třetí (3NF) – je-li ve druhé a zároveň neexistují závislosti neklíčových sloupců tabulky [37] (tzv. tranzitivní závislosti).

[9] Tentýž údaj se vyskytuje na mnoha místech a jeho modifikace vyžaduje provedení příliš mnoha operací [23].

2 BUSINESS INTELLIGENCE

Dnešní podnikové databázové systémy jsou tedy schopny bezpečné a rychlé práce s obrovským množstvím dat, které produkují provozní systémy. Každý výrobní podnik bude shromažďovat údaje z technologických zařízení, firemní administrativy, odbytu a podobně. Každá banka bude evidovat velice podrobné údaje o svých klientech, účtech a transakcích, supermarkety zase data ze skladových karet a elektronických pokladen, mobilní operátoři údaje o zákaznících, frekvenci a délce hovorů atd. Nepředstavitelný objem a neustálý nárůst[10] těchto dat může ilustrovat příklad mobilního operátora T-Mobile, jehož call-centrum (v ČR) přijme denně 40 – 50 tisíc hovorů [6] a o všech těchto hovorech si vede podrobnou evidenci pro marketingové a jiné účely.

Mohlo by se tedy zdát, že má podnik obrovské množství informací, které mu mohou přinést konkurenční výhodu. Záměrně jsem ale uvedl, že v databázích jsou uloženy data (údaje) nikoli informace. A jakým způsobem tyto data transformovat na informace, které budou kvalitní, objektivní, relevantní a rychle dostupné? Jak k nim umožnit přístup manažerům a analytikům, kteří je v dnešním, stále tvrdším konkurenčním prostředí, potřebují pro svá rozhodnutí? Jak zajistit minimální technickou náročnost na manipulaci a zároveň možnost rychle formulovat nové požadavky na další informace odpovídající aktuální situaci na trhu?

2. 1 Proč nevyužít OLTP?

Nejjednodušším řešením by jistě bylo získat informace pro podporu rozhodování přímo z ERP systémů, což je ale z následujících důvodů nevhodné a neefektivní:

- Databáze OLTP jsou v podniku *primárně určeny pro pořizování a aktualizace dat*. Dosahují vysokých výkonů spíše při online transakcích než při složitých analýzách pro podporu rozhodování, které jsou velmi náročné na výkon procesorů. Vzhledem k vytížení těchto systémů by je analytické úlohy buď nadměrně zatěžovaly, nebo by jejich provádění nebylo vůbec možné.
- Podniky mívají svá *data roztroušená v různých zpravidla heterogenních systémech OLTP*, které ani nemusí běžet na stejném operačním systému a tyto data je třeba nejdříve integrovat do jednoho uložiště.
- *Neumožňují manažerům tzv. multidimenzionální pohledy na podniková data*. Manažer může chtít například vidět "jaký zisk přinesl prodej jednoho z výrobků na jedné pobočce za určité časové období". OLTP aplikace nedokáží pružně měnit kritéria (např. sledovat informace o prodeji – v čase, podle zákazníků, produktů, segmentů trhu, poboček, atd.) a umožnit rychlý přístup k agregovaným datům na nejrůznějších úrovních agregace (za podnik, útvar, za všechny zákazníky, skupiny zákazníků i jednotlivé zákazníky) [29].
- V těchto transakčních databázích *obtížné najít příčiny a vysvětlení problémů a závislosti* jednotlivých veličin [25].
- Primární OLTP systémy podniku *neuchovávají historická data*.

2. 2 Historie

Řešení, směřující k odpovědi na otázky z úvodu kapitoly, to znamená k podpoře analytických a manažerských úloh v podnikovém řízení, se začala objevovat už na konci sedmdesátých let minulého století v souvislosti s rozvojem on-line zpracování dat. Průkopníkem v této oblasti byla americká firma Lockheed. V polovině osmdesátých let pak byly publikovány první

[10] Množství dat v podniku se v průměru zdvojnásobí každých pět let [29].

významné práce k tomuto typu aplikací[11]. První komerční produkty, založené na multidimenzionálním uložení a zpracování dat, uvedly v USA na trh v druhé polovině osmdesátých let firmy Comshare a Pilot. Tyto systémy byly označovány jako *EIS* (*Executive Information System*) [29].

Termín *Business Intelligence* zavedl v roce 1989 Howard J. Dresner, analytik společnosti Gartner Group, který jej popsal jako

„sadu *konceptů a metod určených pro zkvalitnění rozhodnutí firmy*“ [29].

Do dnešní doby neexistuje jednotná definice tohoto magického slovního spojení. Já bych rád uvedl definici České společnosti pro systémovou integraci:

„*Business Intelligence je sada procesů, aplikací a technologií, jejichž cílem je účinně a účelně podporovat rozhodovací procesy ve firmě. Podporují analytické a plánovací činnosti podniků a organizací a jsou postaveny na principech multidimenzionálních pohledů na podniková data*“ [49].

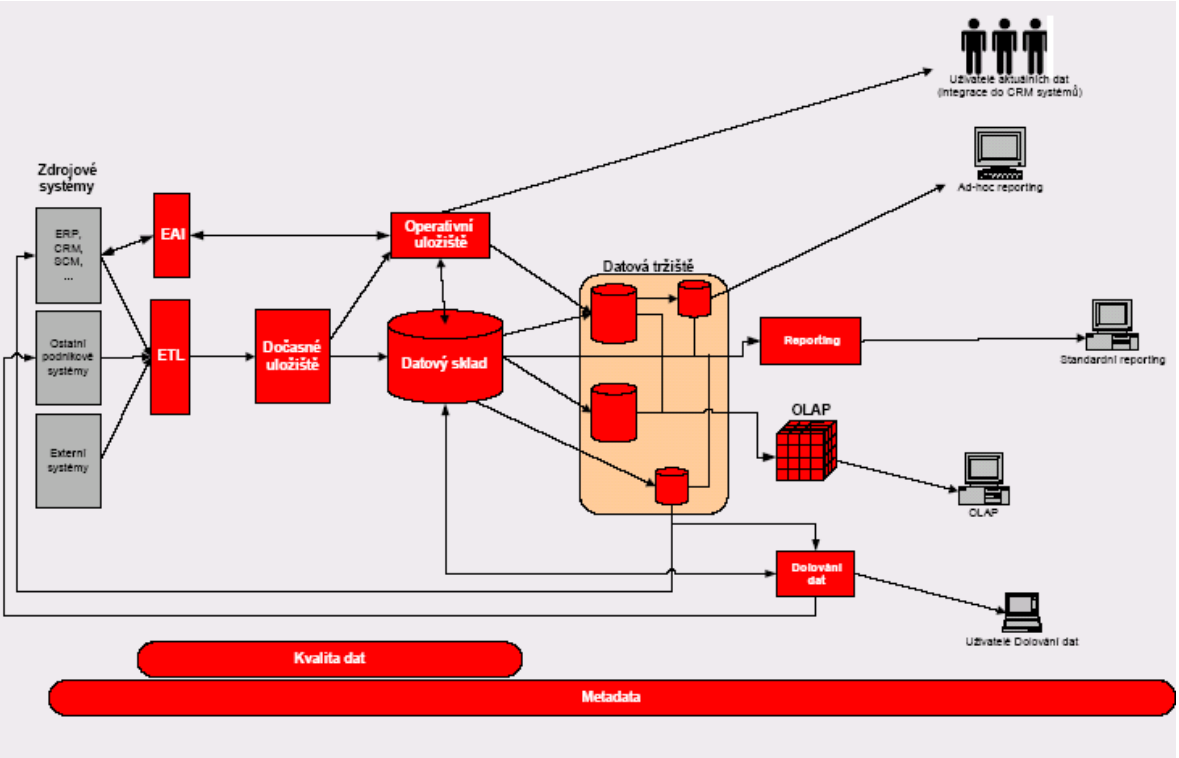
Zjednodušeně by se dalo říci, že Business Intelligence je proces transformace dat z transakčních systémů na informace a převod těchto informací na poznatky (znalosti) prostřednictvím objevování a analýz [10].

2. 3 Architektura BI

Samotné BI řešení se skládá z několika částí, jež se navzájem podporují a doplňují. Tyto části dohromady integrují dostupné datové zdroje a s použitím konkrétní technologie transformují obsah těchto zdrojů dat na požadované výstupy, které se dále používají pro strategické plánování a rozhodování [35]. Jednotlivé části a jejich uspořádání se mohou lišit podle situace a potřeb jednotlivých podniků. Ustálila se ale obecná koncepce architektury BI řešení, která obsahuje následující vrstvy [29]:

- *vrstva pro datovou transformaci,*
- *vrstva pro ukládání dat,*
- *vrstva pro analýzy dat,*
- *vrstva pro prezentaci koncovým uživatelů,*
- *vrstva funkcionální znalosti.*

[11] Např. práce prof. Rockarta *CEO Goes on-line* a další.



Obr. 1 - Hlavní komponenty BI a jejich vazby. Zdroj: [13]

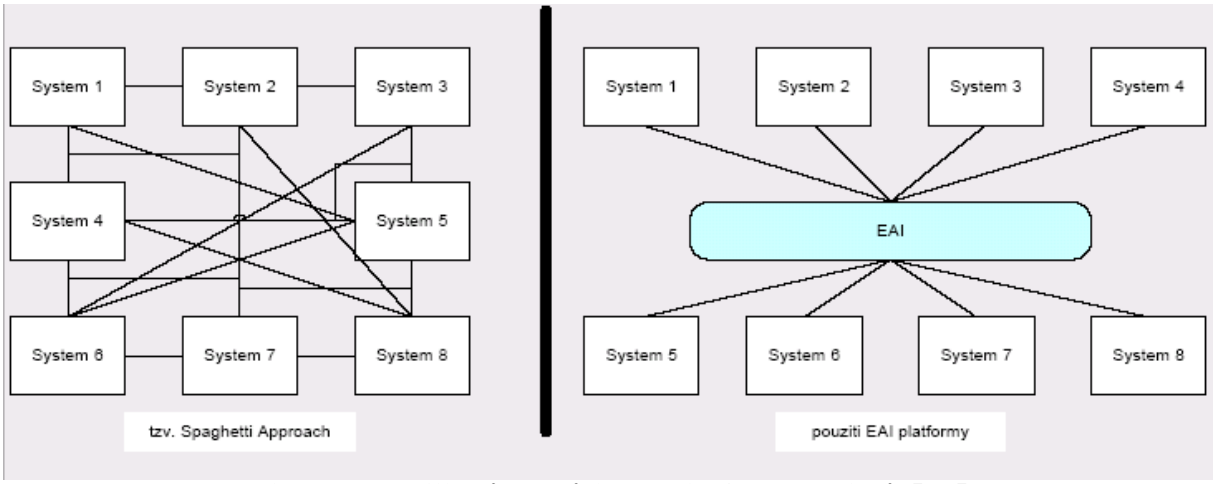
2. 4 Vrstva pro transformaci dat

Úkolem této vrstvy je získat vstupní data z primárních systémů. Nejde zde o pouhé kopírování, protože data jsou získávána z heterogenních systémů, kde jsou uložena v různých formátech a také kvalita těchto dat není vyhovující pro další analytické zpracování. Využívají se zde tzv. *ETL* (*Extraction Transformation Loading*) nástroje. Vzhledem k tomu, že jejich úkolem je přenést (a případně transformovat) data ze zdrojového do cílového systému, mluví se o nich také jako o *datových pumpách*. Podrobněji se jim budu věnovat v kapitole 3.

Dalšími nástroji používanými v této vrstvě jsou *EAI* (*Enterprise Application Integration*), které se sice využívají především v primárních ERP systémech, ale jsou velmi důležité i pro některé BI řešení. V ERP slouží k integraci aplikací a podnikových datových zdrojů za účelem snadnějšího sdílení obchodních procesů a dat[12]. Využívá k tomu XML[13]. Integrace musí být provedena bez nutnosti provádění významných změn v existujících aplikacích a datech [43]. Vzhledem k tomu, že dokáží pracovat v reálném čase, využívají se v BI pro přenos dat do datových uložišť při budování nové generace BI, která umožňuje využít data k analýzám s minimálním zpožděním. Díky EAI také můžeme okamžitě získanou informaci z BI vrstvy zapsat zpět do primárního systému.

[12] V bankovnictví je najdeme například ve funkci propojení mezi primárním systémem, elektronickými službami (Internet banking, GSM banking), call centrem a dalšími produktově orientovanými systémy (platební karty, prodej podílových listů, některé přepážkové služby) [51].

[13] XML (eXtensible Markup Language) je značkovací jazyk pro popis dokumentů obsahujících strukturovaná data. Narozdíl od jazyka HTML se zaměřuje nikoli na jejich vzhled, ale na strukturu. Tvoří tak syntaktický základ pro tvorbu složitějších struktur a vztahů mezi daty [3].



Obr. 2 - Rozdíl při použití EAI platformy. Zdroj: [29]

2. 4. 1 Vrstva sloužící k ukládání dat

K prvnímu ukládání extrahovaných dat dochází v tzv. dočasném uložišti dat *DSA* (*Data Staging Area*) neboli *Stage*. Tato komponenta není v BI řešení vždy, ale je vhodné ji realizovat ze dvou důvodů. Jedním je snížení dopadu provozu BI na výkon neustále vytížených primárních systémů a druhým je možnost dodatečně zde provést některé transformace a konverze, případně některé dopočty. Velmi častá jsou ale řešení, kdy je DSA přesnou kopií relevantních tabulek z primárních systémů, které jsou zde uloženy ve stejné struktuře. V těchto případech se konvertují pouze data, která nejsou v databázových formátech (např. textové soubory) a veškeré další transformace probíhají při převodu dat z DSA do trvalého uložiště, kterým je *datový sklad*.

V každém případě data v DSA neobsahují historii (přenášejí se pouze aktuální data ze zdrojových systémů), nejsou konzistentní (nejsou kontrolována proti externím číselníkům či ostatním datům v datovém skladu) a mění se (při každém snímku se berou pouze data, která ještě nebyla zpracována a po jejich zpracování a přenosu do dalších vrstev jsou odstraněna) [29]. Do DSA mají přístup pouze členové vývojového týmu, neposkytuje žádné prezentační funkce.

Dalším nepovinným uložištěm dat je *ODS* (*Operational Data Store*). Tato komponenta by se dala přirovnat k operační paměti počítače, protože stejně jako ona obsahuje aktuální data. V této analogii by pak samotný datový sklad byl jakýmsi pevným diskem počítače[14]. ODS obvykle obsahuje relativně malý objem dat a stejně jako u DSA jsou data neustále aktualizována. Je využívána zejména pro jednoduché dotazy nad malým množstvím aktuálních analytických dat [29]. Výsledek těchto dotazů by měl přijít v intervalu 2 – 3 sekundy [19]. ODS může vzniknout buď jako derivace již existujícího datového skladu [29] nebo může být plněna integračními a transformačními aplikacemi přímo z primárních systémů.

Hlavním uložištěm dat je *datový sklad* (*Data Warehouse, DW*). Ačkoli by se podle názvu mohlo zdát, že slouží pouze jako odkladiště, není pravda. Naopak pokud je datový sklad navržen správně, uložená data jsou intenzivně využívána k podpoře rozhodování. Jedná se o dlouhodobé uložiště, kam jsou přenášena po jednotlivých *dávkách* (*loadech*) použitelná data z primárních systémů. Datový sklad se vyznačuje jinou architekturou než klasické OLTP systémy. Vyplývá to z odlišných požadavků a funkce, kterou DW zastává v architektuře podnikových informačních systémů. Zatímco do primárních systémů se během pracovního

[14] Za předpokladu, že by se na něj pouze ukládalo a uložená data by nebyla příliš upravována ani mazána.

dne velmi často zapisují nové údaje (např. přidání nové objednávky) a aktualizují již uložené údaje, do datového skladu se zapisuje většinou přes noc a v rámci pracovního času manažerů se z něj pouze čte. Je proto optimalizován pro maximální možné zefektivnění provádění dotazů a výpočtů nad již uloženými daty. Tabulky v DW proto nemusí splňovat normální formy, připouští se i určitá *redundantnost* (vícenásobné uložení stejných dat) a také nižší *granularita* (detailnost) uchovávaných dat [26].

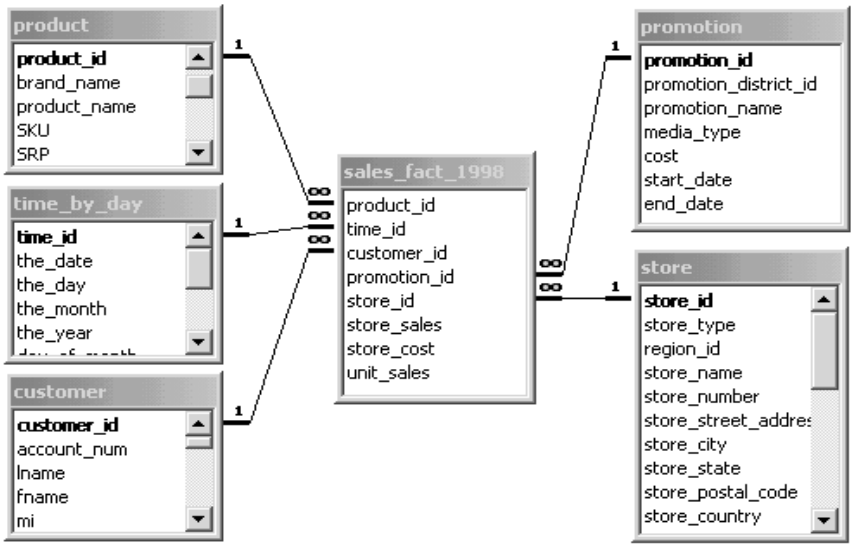
V datovém skladu jsou použity dva základní typy tabulek:

- *tabulky faktů,*
- *dimenzionální tabulky.*

Tabulka faktů obsahuje numerické měrné jednotky obchodování. Je zde uložen např. počet prodaných kusů u maloobchodu, průměrný denní zůstatek a objem transakcí u bank či objem pohledávek pojišťovny. Faktová tabulka bývá v DW tabulkou s největším objemem dat.

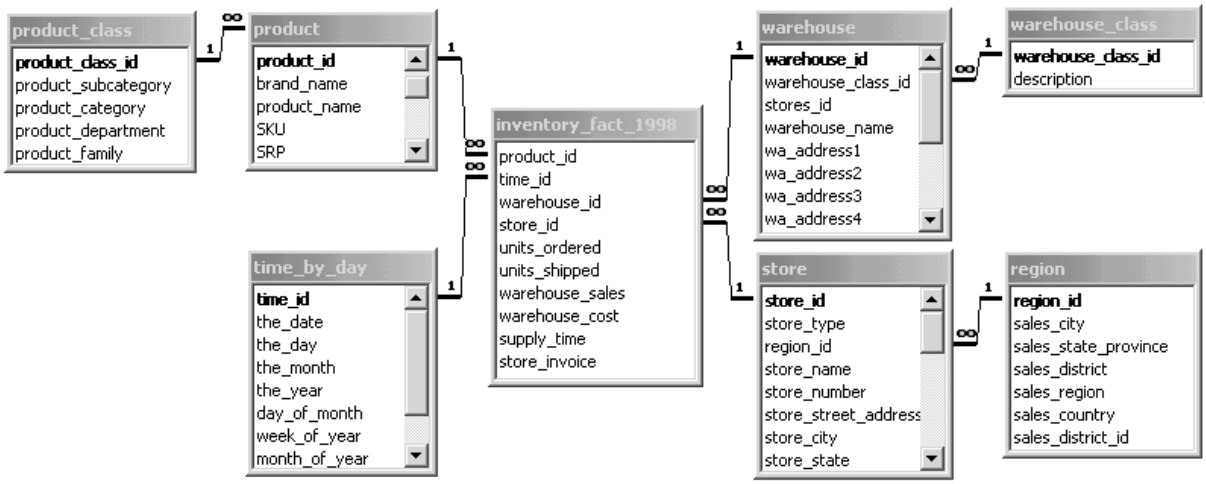
Dimenze pak obsahují textové popisy obchodování. Stanovují obsah faktů a umožňují jejich interpretaci. Často mají hierarchické uspořádání a typická dimenze je např. dimenze "Region", kde může být hierarchie: Stát – Kraj – Město. V DW také najdeme tzv. časovou dimenzi, která obsahuje hierarchicky uspořádané údaje podle kalendářních zvyklostí a jejím úkolem je zajistit historický pohled na data.

Tabulky faktů a dimenzí je možné uspořát do dimenzionálního modelu dvěma způsoby. Prvním je tzv. *Star schema* (*hvězdicové schema*). Faktová tabulka zde obsahuje cizí klíče, vztahující se k primárním klíčům v dimenzionálních tabulkách a je zde plně normalizovaná, zatímco dimenze denormalizovány. Takové schéma pak poskytuje vysoký dotazovací výkon, ale za cenu redundantního uložení dat.



Obr. 3 - Star schema. Zdroj: [17]

Druhou možností je *Snoflake schema* (*schéma sněhové vločky*), kde některé dimenze tvoří více relačně svázaných tabulek. Redukuje se tím redundance, ale v dotazech je třeba spojovat více tabulek, což snižuje dotazovací výkon [44].



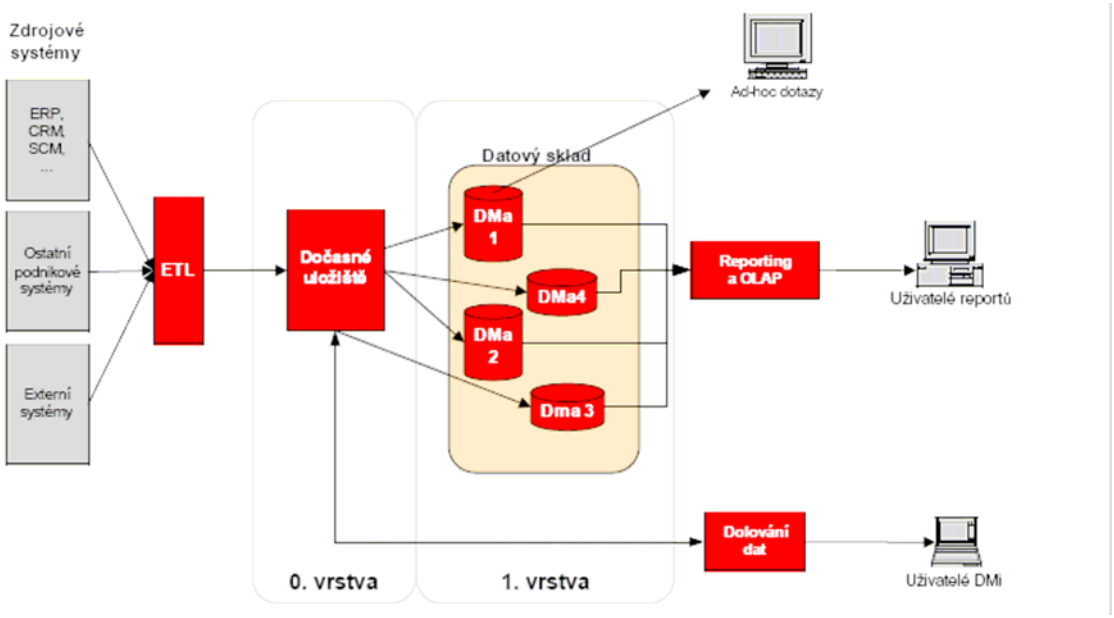
Obr. 4 - Snowflake schema. Zdroj: [17]

Rozeznáváme dvě základní architektury datových skladů [7]:

- *nezávislé Data marty* (“Bus architecture”),
- *konsolidovaný datový sklad.*

Koncept architektury nezávislých datových tržišť byl popsán v 80. letech Ralphem Kimballem[15]. Jeho princip spočívá v relativně nezávislém vytváření jednotlivých datových tržišť pro specifické útvary podniku (divize, oddělení, pobočky, závody). Každé takové tržiště je typicky kompletně životaschopné, tzn. obsahuje veškeré vrstvy a komponenty, které umožní získat data z primárních systémů, zpracovat je, uložit v datovém tržišti, případně analyzovat pomocí OLAP aplikací či Data Mining komponent a prezentovat je uživateli. Vzhledem k vývoji požadavků na řešení BI byl původní koncept R. Kimballem v 90. letech přepracován a vznikla tzv. sběrnicová architektura (v anglickém originále "bus architecture"). Rozdíl oproti předchozímu chápání je pouze ve snaze budovat jednotlivá nezávislá datová tržiště integrovaně. Integračním prvkem jsou tzv. *sdílené dimenze*, tedy dimenzionální tabulky, které jsou opakovaně použity v různých datových tržištích [28]. Původní návrh totiž vyžadoval, aby každé oddělení mělo vlastní data, definici pojmů, historii dat a provádění vlastní aktualizace dat [22]. To vedlo k redundanci v datech a odlišnému pohledu na ně.

[15] Podle Kimballa „...není datový sklad ničím jiným než sjednocením všech data martů…“ [20].

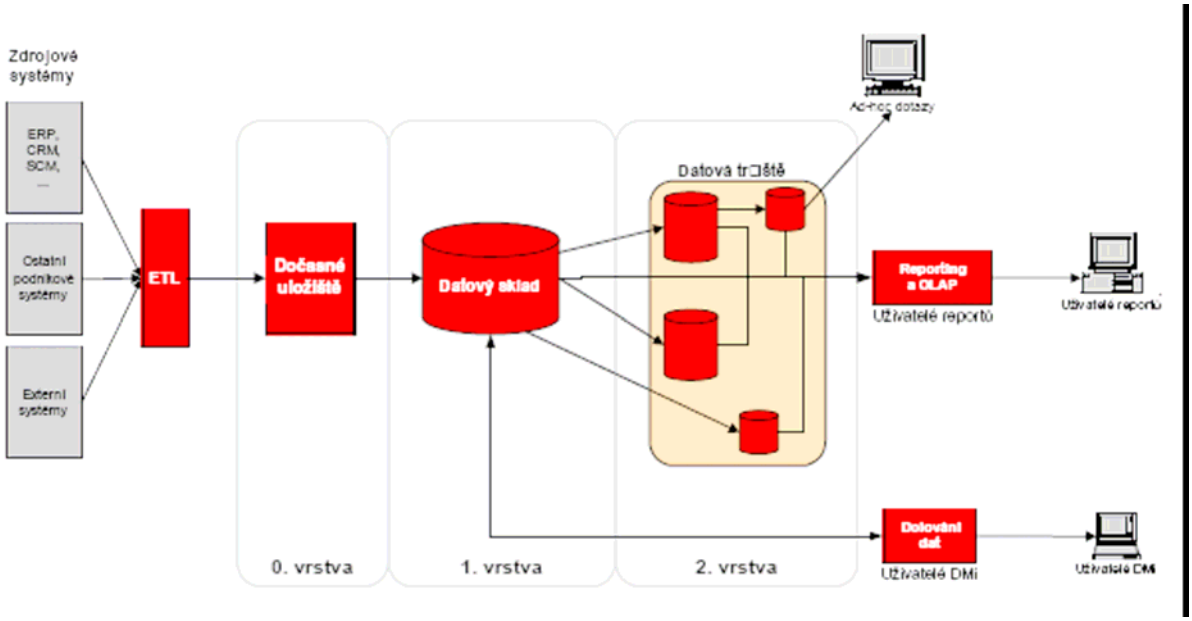


Obr. 5 - Architektura nezávislých datových tržišť. Zdroj: [29]

Kimballova myšlenka se stala díky rychlé implementaci data martů a nižších počátečních investic velmi populární. Na druhé straně, je potřeba vyvinout větší úsilí pro zajištění celkové integrace BI řešení a následné budování jednotných vyšších vrstev může být finančně i časově velmi náročné. Tento typ DW je vhodné budovat, pokud je potřeba vybudovat rychle řešení pro několik nezávislých oddělení, přičemž se do budoucna nečeká jejich integrace [28].

S druhým způsobem budování datového skladu přišel Bill Inmon. Jeho definice datového skladu zní:

„*Datový sklad je kolekce sjednocených, předmětově orientovaných databází, navržených za účelem poskytovat informace potřebné pro rozhodování*“ [18].



Obr. 6 - Architektura konsolidovaného datového skladu. Zdroj: [29]

Podle Inmonovy definice je datový sklad *sjednocený*. To znamená, že spojuje data z mnoha různorodých provozních systémů a poskytuje na ně integrovaný pohled [16]. Jedná se o údaje typu evidence zákazníků, které mohou být uložené v různých OLTP systémech. A je téměř jisté, že jinak bude vypadat evidence pro účely účetnictví a jinak pro marketingové oddělení. Podnik může navíc čerpat data i z externích systémů. V datovém skladu je potřeba mít uložena zákaznická data konzistentně v jednotných formátech a je třeba zabránit duplicitám (např. že "Jan Novák z ulice Stanislavova 5" z jedné databáze je stejná osoba jako "Honza Novák, Stanislavova 678/5" z jiné databáze a oba budou tvořit v datovém skladu jeden záznam "Novák, Jan, Stanislavova 678/5"). To zajišťuje etapa ETL.

Další charakteristikou DW je podle Inmona *předmětová orientace*. Tradiční provozní systémy jsou zaměřeny na potřeby oddělení či divizí a vytvářejí velmi kritizované tzv. *cylindrické systémy*. S příchodem tzv. *zpětného inženýrství*[16] obchodních procesů začaly podniky využívat procesně zaměřené týmy a pracovníky pro konkrétní případy. Moderní provozní systémy přenesly pozornost na provozní požadavky celého obchodního procesu a zaměřily se na podporu celého obchodního procesu od začátku do konce. Datový sklad přinesl tradiční informační pohledy na celopodnikové subjekty, jako jsou zákazníci, prodeje či zisky. Tyto subjekty určují jak hranice organizační, tak procesní a pro poskytnutí kompletního obrazu vyžadují informace z více zdrojů [16].

Datový sklad, založený na Inmonově architektuře, je možné vybudovat buď *jednorázově* nebo tzv. *přírustkovou metodou*. Přírustkový přístup je sice z hlediska vývoje historicky nejmladším, ale v současnosti nejčastěji implementovaným, především díky jeho dobré finanční kontrolovatelnosti a sledování ROI[17].

2. 4. 2 Vrstva pro analýzy dat

První analytickou vrstvou je *reporting*. Umožňuje přístup k informacím v datovém skladu uživatelům, kteří nevládnou dotazovacím jazykem. Díky reportingovým nástrojům jsou výsledky dotazů k dispozici manažerům a analytikům v rozličných formátech, jako je např. HTML, MS Excel, PDF, atd.

Kromě standartního reportingu, kdy jsou v určitých časových periodách spouštěny předpřipravené reporty (pevné tabulky, grafy), existuje i tzv. *Ad-hoc reporting*, podporující jednorázovou formulaci specifických dotazů.

Na pokročilé a dynamické analytické úlohy je zaměřena vrstva *OLAP* (*Online Analytical Processing*). Termín OLAP zavedl[18] Dr. Codd, tvůrce relačního databázového modelu, aby mohl popsat technologii, která by pomohla překlenout mezery mezi využitím osobních počítačů a řízením podnikových dat [25]. Praktičtější a srozumitelnější přístup nabízí tzv. pojetí *FASMI* [34]:

- *Fast* – podle nezávislého výzkumu, provedeného v Holansku, uživatelé stisknou „Ctrl+Alt+Del", pokud neobdrží výsledky do 30 sekund. OLAP aplikace tedy musí odpovědět na jednoduché dotazy do 1 sekundy a drtivá většina dotazů by měla trvat méně než 20 sekund. Rychlost odpovědi navíc nesmí záviset na množství dat.
- *Analysis* – spojení obchodní logiky s daty. Aplikace by měly být dostatečně flexibilní.
- *Shared* – všichni uživatelé musí mít k dispozici stejná data (databáze musí být konzistentní) a ne každý uživatel má přístup ke všem datům.

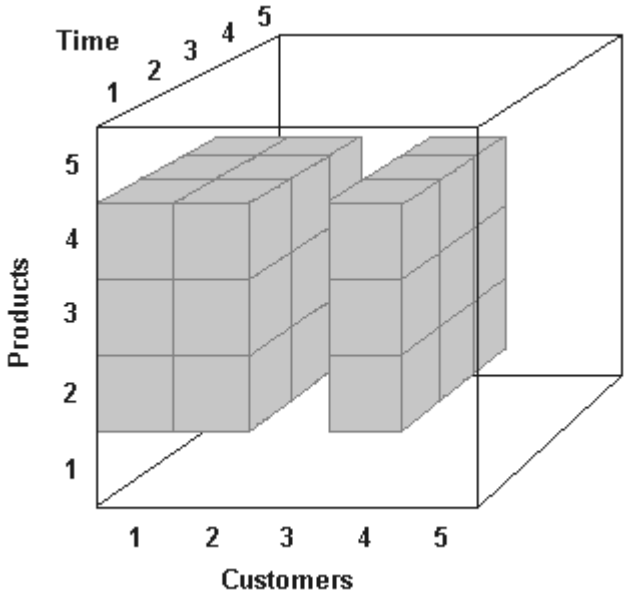
[16] Zpětné inženýrství je definováno jako proces analýzy předmětného systému s cílem identifikovat komponenty a jejich vzájemné vazby a/nebo vytvořit reprezentaci systému v jiné formě nebo na vyšší úrovni abstrakce [47].

[17] Finanční ukazatel návratnosti investic (Return of investment).

[18] Bylo to v roce 1993 v dokumentu „Providing OLAP to User-Analyst: An IT Mandate". Dr. Codd zde také zavedl 12 základních pravidel OLAPu.

- *Multidimenzional* – klíčový požadavek. OLAP poskytuje uživateli multidimenzionální pohled na data.
- *Information* – zajímáme se o to, jaký objem informací je schopna daná technologie zpracovat, nikoli kolik GB zabere jejich uložení.

Dalo by se říci, že stejně jako je tabulka základním stavebním kamenem relačního modelu, u multidimenzionálního modelu je to *krychle*. V jednotlivých buňkách jsou hodnoty neboli *ukazatele*. To jsou veličiny, které sledujeme. Strany krychle tvoří prvky jednotlivých dimenzí. Kromě atomických prvků krychle většinou obsahuje i *agregace*. Příklad krychle obsahující tři dimenze (zákazník, produkt, čas) je vidět na následujícím obrázku.



Obr. 7 - OLAP. Zdroj: [17]

Multidimenzionalita spočívá v možnosti dimenze libovolně kombinovat (*slice&dice, crosstabing*) a sledovat dimenze na různých úrovních agregace (*drill-up*, *drill-down, drill-across*). Lze také provádět aritmetické i množinové operace nad buňkami kostky, vyhledávat extrémy a použít agregační a statistické funkce. Na rozdíl od běžného fyzikálního modelu může krychle obsahovat více dimenzí[19].

Krychle mohou být uloženy na serveru jako:

- *MOLAP* – data jsou získána z datového skladu a mechanismus MOLAP je uloží ve vlastních strukturách a sumářích. Během procesu se spočítá tolik předběžných výsledků, kolik je z časového a technického hlediska možné. Údaje jsou uloženy jako dopředu vypočítaná pole. To umožňuje dosažení maximálního výkonu vzhledem k dotazům uživatelů, nevýhodou je ale vysoká míra redundance, protože data jsou uložena jak v relační, tak multidimenzionální databázi [25].
- *ROLAP* – data jsou uložena v relační databázi a uživateli je předkládán pouze multidimenzionální pohled. Nevzniká redundance, dotazovací výkon je ale výrazně nižší.

[19] U MS SQL Server 2000 je to až 64 dimenzí, verze 2005 není omezena počtem dimenzí v jedné krychli.

- *HOLAP* – neboli hybridní OLAP ukládá údaje do relačních a agregace do multidimenzionálních struktur. Navíc využívá paměť cache pro dotazování. Eliminuje tak nevýhody obou předchozích způsobů uložení.

Ačkoli metody OLAP umožňují udržovat uživatelům přehled o okamžité situaci podniku či rychlou přípravu konsolidovaných finančních reportů, existuje mnoho úloh, na které tyto postupy nestačí. Je potřeba využít tzv. *data miningu* neboli dobývání znalostí z databází, což podniku poskytne možnost, za využití speciálních algoritmů, automaticky objevovat strategické informace v datech[20]. Rick Fayaad definuje data mining jako:

"*Netriviální proces zjišťování platných, neznámých, potenciálně užitečných a snadno pochopitelných závislostí v datech*" [9].

Tento proces je netriviální, protože k vytváření prediktivních nebo deskriptivních modelů užívá sofistikovaných statistických metod a metod umělé inteligence, jako jsou [40]:

- *Lineární regrese* - kvantifikuje závislost mezi dvěmi spojitými proměnnými: závislou, neboli proměnnou, kterou se snažíme předikovat a nezávislou, tedy prediktivní proměnnou. Tímto modelem můžeme sledovat např. vztah mezi tržbami a reklamou.
- *Logistická regrese* - je velmi podobná lineární, hlavním rozdílem je, že závislá proměnná není spojitá, je diskrétní neboli kategorická. Pomocí ní lze (s určitou pravděpodobností) předvídat diskrétní akci, např. odezvu na nabídku nebo nesplácení půjčky.
- *Neuronové sítě* - nevychází ze statistického rozdělení, jsou modelovány podle funkce lidského mozku, založeného na systému neuronů. Celý proces je zaměřen na přijímání informací a poučení se z každé zkušenosti. Používají se především pro prediktivní modely, kde dokáží (na rozdíl od regrese) vystihnout nelineární vztahy.
- *Genetické algoritmy* - řídí se evolučním procesem „přežití nejpřizpůsobivějšího". Větší počet modelů je v sérii iterací upravován, dokud není nalezen nejlepší model pro danou úlohu. Jedná se o vynikající, ale na počítačové zpracování velmi náročnou, cestu k nalezení vhodného modelu.
- *Klasifikační stromy* - účelem je roztřídit data do odlišných skupin či větví, které vytvoří nejsilnější separaci hodnot závislé proměnné. Každá skupina se shodnými vlastnostmi pak tvoří jeden list, který odpovídá určitému segmentu definovanému předešlými uzly.

Typické oblasti využití data miningu [39]:

- *Churn attriction* (predikce odchodu zákazníků) - prediktivní model, získaný analýzou dat o zákaznících, lze použít pro plánování akcí, jimiž lze zabránit odchodu (dobrých) zákazníků.
- *Fraud detection* (detekce pojistného, úvěrového a daňového podvodu) - pomocí prediktivního modelování (nejčastěji neuronové sítě) či shlukové analýzy lze odhalit podezřelé chování a platební transakce (i praní špinavých peněz).
- *Segmentace zákazníků* - rozdělení databáze zákazníků na segmenty, díky kterému se zvýší efektivita marketingových kampaní. Využívá se typicky v bankovnictví, pojišťovnictví, telcu nebo u velkých obchodních řetězců.
- *Vyhodnocení marketingových kampaní* - pomocí prediktivního modelu odezvy, získaného na základě dat ze vzorku zákazníků, jsou z databáze vybráni zákazníci s největší pravděpodobností odezvy, na které je pak cílena kampaň.
- *Analýza produktů* - umožňuje definovat komplementární produkty pro jednotlivé segmenty zákazníků.

[20] Je třeba dodat, že tato musí být na data mining připravena. Data miningové modely se budují nad datovým skladem, který obsahuje data v požadované kvalitě.

- *Analýza chování zákazníků* - na základě historických dat predikuje vývoj na trhu.
- *Cluster detection* (shluková analýza)- vytváří modely identifikující datové záznamy, které jsou si navzájem podobné. Využívá se i v analýzách nákupního košíku (market basket analysis).

2. 4. 3 Prezentační vrstva

Obsahuje nástroje pro konečné uživatele a zajišťuje jejich komunikaci s ostatními komponentami BI řešení. Jde zejména o sběr požadavků na analytické operace a následnou prezentaci výsledků. Nástroje mohou být dvojího typu:

- *Tenký klient* – pro přístup k údajům je použito webové rozhraní a standardní prohlížeč. Kapacita klientského počítače je využita pouze pro zobrazení, veškeré zpracování dat probíhá na serveru.
- *Tlustý klient (rich klient)* – klientská aplikace běžící na lokálním počítači (např. EIS systémy). Používá naplno jeho hardwaru a operačního systému, stále ale využívá služeb nějakého serveru.

2. 4. 4 Vrstva funkcionální znalosti

Vrstva funkcionální znalosti neboli oborové znalosti (know-how) zahrnuje oborovou znalost a tzv. best practices nasazování řešení BI pro konkrétní situaci v organizaci [30].

3 ETL

Jak jsem již uvedl, nástroje a metody ETL jsou velmi důležitou součástí Business Intelligence řešení. Roli ETL by dokonce bylo možné označit za klíčovou pro úspěch celého projektu, protože se jedná o mechanismus získávání dat z provozních systémů podniku, jejich následné zpracování a poskytnutí aplikacím pro podporu rozhodování [45]. Klíčová role ETL vyplývá z toho, že pokud BI systémy obdrží chybná výchozí data, nesprávné budou i všechny následné analýzy, celý komplikovaný proces bude zbytečný a pokud se podle takovýchto výsledků analýz budou manažeři rozhodovat, napáchá více škody než užitku. Proto je také procesu ETL věnována náležitá pozornost, podle již zmiňované, renomované společnosti Gartner Group, až 45% času při budování datového skladu [51]. Jiné zdroje [45] uvádějí, že problematika ETL je natolik komplikovaná, že může představovat dokonce až 70% nákladů na budování systémů na podporu rozhodování, a to jak finančních, tak časových i lidských. Proces ETL lze rozdělit do následujících fází:

- *Extrakce* – z primárních systémů se extrahují relevantní data.
- *Transformace* – transformují se do požadované podoby.
- *Loading* – přemístí (načtou) se do datového skladu.

Ne vždy ale proces probíhá v tomto pořadí, stále více v oblibě jsou systémy *E-LT* (*Extract – Load & Transform*) , které jsou v některé literatuře [8] označovány za třetí generaci[21] ETL nástrojů.

3. 1 Extrakce

Úkolem první fáze ETL je vyextrahovat z primárních systémů data, která jsou zajímavá z hlediska analýzy. Tento proces komplikuje fakt, že data mohou být uložena v nehomogenních operačních prostředích, hardwarových platformách (PC, mainframe, Mac, …), operačních systémech (Windows, Unix, Linux, Sun, Solaris, …), serverových databázových systémech (MS SQL Server, Oracle, Informix, DB2, MySQL, PosgreSQL, Sybase, …), desktop databázích (MS Access, FoxPro, …), informačních systémech (SAP, Baan, Abra, PeopleSoft, …), archivních a externích systémech (Internet, zakoupené údaje o zákaznících, …). Zdrojový systém může být jakákoli aplikace nebo uložiště dat, která vytváří nebo uchovává data [13]. Je potřeba si uvědomit, že tato data budou uložena v rozličných formátech. Komunikuje se různými způsoby (ODBC[22] rozhraní, nativní drivery, použití flat-files[23]).

3. 2 Transformace

Pokud se podaří načíst data z externích systémů, následuje fáze transformací. Je totiž potřeba zajistit požadovanou *datovou kvalitu*. Ta zajišťuje, že data odpovídají realitě [7]:

- *úplnost* – jsou identifikována a ošetřena data, která z různých důvodů chybí nebo jsou nepoužitelná,

[21] Za první generaci je označováno ETL, kde integrační nástroje generují nativní kód (velmi často je použit COBOL) pro operační systém platformy, na které se budou data zpracovávat. Druhá generace jsou tzv. proprietární ETL nástroje. Takovéto nástroje mají proprietární engine, který je umístěn mezi cílem a zdrojem a řídí veškeré transformační procesy. Tímto způsobem se řešení vypořádá s problémy způsobené rozdílnými programovacími jazyky, používanými v různých platformách. Všechna data, pocházející z různých zdrojů, jsou zpracovávána řádek po řádku [8].

[22] Open Database Connectivity – sada ovladačů pro obecný přístup k databázovým strojům od společnosti Microsoft.

[23] Flat-file je zdroj na nerelační bázi, typicky textový soubor vygenerovaný nějakým databázovým strojem, kde jsou data v řádcích a k rozlišení sloupců je použit dohodnutý oddělovač (delimiter). Tímto oddělovačem bývá čárka nebo tabulátor.

- *soulad* – jsou identifikována a ošetřena data, která nejsou uložena ve standardním formátu,
- *konzistence* – jsou identifikována a ošetřena data, jejichž hodnoty reprezentují konfliktní informace,
- *přesnost* – jsou identifikována a ošetřena data, která nejsou přesná nebo jsou zastaralá,
- *unikátnost* – jsou identifikovány a ošetřeny záznamy, které jsou duplicitní,
- *integrita* – jsou identifikována a ošetřena data, která postrádají důležité vztahy vůči ostatním datům.

Podle výzkumu [2] agentury Market, realizovaného pro společnost Adastra, považuje 57% společností vliv nekvalitních informací na fungování organizace za kritický nebo velmi vážný.

K zajištění datové kvality se využívá transformací několika typů:

- *změna formátu,*
- *odstranění duplicitních záznamů,*
- *nahrazení chybějících hodnot,*
- *rozdělení položek,*
- *sloučení položek,*
- *odstranění nejednoznačnosti a nesrozumitelnosti údajů,*
- *zajištění referenční integrity,*
- *doplnění chybějícího časového údaje,*
- *doplnění agregovaných hodnot,*
- *detekce a oprava nejasných číselníků.*

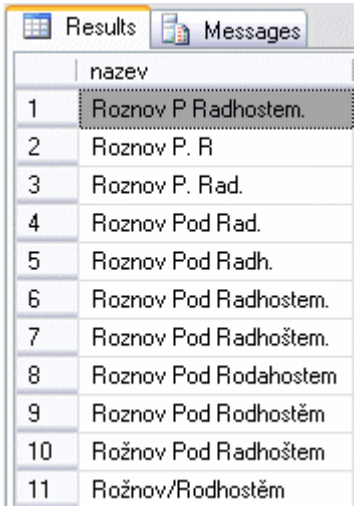
3. 2. 1 Změna formátu

Data, která jsou v primárních systémech uložená v různých formátech, jsou konvertována do jednotného databázového formátu. Typickým příkladem je např. datum a čas, který může být zapsán a uložen mnoha způsoby. Pro správné fungování datového skladu je ale nutné, aby byl uložen jednotně.

3. 2. 2 Odstranění duplicitních záznamů

Primární systémy často obsahují stejná data uložená na různých místech (typicky databáze zákazníků, kterou si bude vést jak finanční, tak marketingové oddělení firmy) a úkolem ETL je tato data integrovat a vytvořit tzv. *jednotnou verzi pravdy*[24]. K určení, jestli dva podobné záznamy z různých zdrojů lze sloučit do jednoho, je často třeba použít sofistikovaných metod a specializovaných nástrojů a celý proces bývá časově náročný. Některé potenciální duplicity nelze zpracovat automaticky a je nutné označit je jako výjimky či podezření a vyhodnotit je ručně. Řešením je také označit jeden zdroj za řídící. Samozřejmostí je odstranění duplicit z jednoho zdroje. Pro ilustraci uvádím příklad z praxe [7], který ukazuje, že jen jeden název města lze napsat více než 100 způsoby. Část výpisu ukazuje Obr. 8. Duplicity zde byly vytvořeny lidským faktorem, najdeme zde nesprávné názvy, překlepy, atd.

[24] Výstupy z různých systémů mohou být obtížně porovnatelné a datový sklad by měl definovat společné principy, pravidla a algoritmy, kterými jsou zpracována veškerá data [52].



Results | Messages

	nazev
1	Roznov P Radhostem.
2	Roznov P. R
3	Roznov P. Rad.
4	Roznov Pod Rad.
5	Roznov Pod Radh.
6	Roznov Pod Radhostem.
7	Roznov Pod Radhoštem.
8	Roznov Pod Rodahostem
9	Roznov Pod Rodhostěm
10	Rožnov Pod Radhoštem
11	Rožnov/Rodhostěm

Obr. 8 - Příklad duplicity údajů

3. 2. 3 Nahrazení chybějících hodnot

Ještě větším problémem než duplicitní záznamy jsou chybějící hodnoty. Jedná se o sloupce relačních databází obsahující `NULL` hodnoty nebo prázdné řetězce. Někdy je možné tyto hodnoty nahradit z jiných zdrojů, jindy je nutné je nahradit vhodnými příznaky, např. „`#neuvedeno`“ nebo „`#neznámo`“.

3. 2. 4 Rozdělení položek

Může se stát, že datové položky, načítané do datového skladu, bude nutno rozdělit do více atributů. Může se jednat např. o adresy zákazníků, čerpané z externích systémů, kde mohou být uloženy ve formátu: „ulice, čp, město, psč“. Pro účely datového skladu bude ale nutné tyto textové hodnoty rozdělit do samostatných atributů [16].

3. 2. 5 Sloučení položek

Někdy je naopak vhodné sloučit několik položek z primárních systémů do jednoho atributu skladu.

3. 2. 6 Odstranění nejednoznačnosti a nesrozumitelnosti údajů

Velmi často nastává situace, kdy musíme při transformaci řešit nejednoznačnost údajů. Jak ukazuje Obr. 9, údaje o pohlaví zákazníka mohou být uloženy různým způsobem [25].

	id	jmeno	prijmeni	pohlavi
1	1	Tomáš	Gottwald	Male
2	2	Martin	Kraft	Man
3	3	Petra	Neveselá	Female
4	4	Jaroslava	Nováková	F
5	5	Jára	Cimrman	male

	id	jmeno	prijmeni	pohlavi
1	1	Tomáš	Gottwald	M
2	2	Martin	Kraft	M
3	3	Petra	Neveselá	F
4	4	Jaroslava	Nováková	F
5	5	Jára	Cimrman	M

Obr. 9 - Nejednoznačnost údajů (vlevo), vpravo požadovaný stav

Dalším problémem může být, že v primárních systémech se často vyskytují různé generované hodnoty, které nejsou konečnému uživateli BI srozumitelné. Takovéto hodnoty je nutné v transformačním procesu nahradit vhodnými příznaky, kterým budou uživatelé analýz

rozumět. Dále je potřeba sjednotit terminologii všech oddělení firmy tak, aby jedné entitě v realitě odpovídal jednotný název.

3. 2. 7 Zajištění referenční integrity[25]

V údajích jsou skryty různé vztahy, např. master-detail, organizační struktura firmy, hierarchická struktura zaměstnanců a podobně [25]. Údaje jsou ale dynamické, organizační struktura se může měnit, což nebývá v transakčních systémech zaznamenáno. Tyto změny[26] je nutné zaznamenat v datovém skladu a zajistit jeho uživatelům možnost prohlížet záznamy podle staré i nové organizační struktury. Správně navržený ETL proces by se měl vypořádat i s neúplnou nebo porušenou referenční integritou a o chybách informovat administrátora a umožnit opravu buď na úrovni primárního systému nebo přímo datového skladu.

3. 2. 8 Doplnění chybějícího časového údaje

V mnoha systémech, ze kterých jsou čerpána data do datového skladu, se neeviduje datum a čas. Vzhledem k tomu, že v datovém skladu hraje časový údaj klíčovou roli, je nutné zajistit jeho přítomnost před zavedením nebo ho určit a přidat při zavádění dat [25].

3. 2. 9 Doplnění agregovaných hodnot

Také agregace mohou být předpřipraveny pro nahrání do datového skladu a jedná se o alternativu k nahrávání pouze atomických (základních) dat a vytváření agregačních záznamů založených na atomických datech skladu [16].

3. 2. 10 Detekce a oprava nejasných číselníků

Databáze velkých firem obsahují velké množství číselníků, které transakční systém potřebuje ke svému fungování. Takovým číselníkem může být např. centrální číselník zboží. Při zavádění číselníků do datových skladů může docházet k situaci, že jednomu druhu skladové položky (zboží) odpovídá větší počet kódů v převzatých datech [51], což má za následek chybné výsledky analýz. Správně navržené ETL musí umožnit detekci i zpětnou opravu takovýchto chyb.

3. 3 Loading

Poslední fází ETL je fáze načítání dat do dočasného úložiště (Stage vrstva) nebo přímo do datového skladu. Tento přenos by měl být co nejvíce automatizovaný, plánovaný a zahrnuje následující operace [16]:

- *blokování indexace skladu* – pro zvýšení výkonu je blokována okamžitá reindexace, která by probíhala po vložení každého záznamu,
- *generování klíčů* – klíče ve zdrojových systémech zpravidla neodpovídají klíčům v DW, proto jsou při nahrávání dimenzí generovány tzv. umělé klíče, které jsou pak použity jako primární klíč pro nahrávání faktů a tím je zajištěna referenční integrita skladu (klíče je možno generovat i před samotným nahráním do DW),
- *výpočet agregací* – pokud již nejsou předpřipraveny,
- *reindexace skladu* – po skončení loadu pro zvýšení dotazovacího výkonu.

Při prvním naplnění datového skladu historickými daty je celý proces časově velmi náročný, další plnění pak probíhají periodicky (např. každou noc) a načítají se pouze data vyprodukovaná primárními systémy v intervalu od předcházejícího loadu. Některé speciální typy analytických úloh však vyžadují, aby vyhodnocení dat proběhlo v co nejkratším čase,

[25] Referenční integrita definuje vztahy mezi různými sloupci různých tabulek relační databáze. Splnění podmínek referenční integrity lze zajistit pomocí definice cizího klíče. Každá hodnota cizího klíče musí odpovídat hodnotě rodičovského klíče [50].
[26] Dimenze, podléhající těmto typům změn nazýváme *SCD (Slowly Changing Dimensions)* [7].

poté co došlo k transakci nebo jiné změně v datech. Typickým příkladem je detekce podvodů při používání platebních karet nebo telekomunikačních služeb [51]. V takových případech se, jak jsem již uvedl, používají nastroje EAI.

3. 4 Metadata

Metadata v etapě ETL popisují strukturu (názvy tabulek, sloupců, datové typy, sémantiku) jak zdrojových dat, tak dat v cílové databázi a definují obchodní pravidla pro transformaci dat. Jsou využívána pro automatizaci některých částí projektu, včetně transformací, kde redukují nebo zcela eliminují práci programátorů a snižují chybovost transformací [51].

3. 5 Ošetření chyb v etapě ETL

Proces ETL nemusí vždy proběhnou úspěšně, naopak při načítání dat dochází často k řadě chyb. Problémy mohou být se spolehlivostí úložiště dat (HW problém), může dojít k výpadkům spojení, zdroje údajů se mohou měnit (např. při upgradu primárních systémů, pokud změny nejsou evidovány v metadatech) [25]. Správně navržený ETL systém by měl všechny chyby zaznamenávat do chybového žurnálu [51]. Některé je možné opravit automaticky, ale na chyby závažnějšího charakteru by měl být (např. emailem) upozorněn administrátor, který pak zajistí jejich manuální opravu. Oprava chyby by měla být provedena v systému, kde se vyskytla, což není bohužel vždy možné a je tedy nutné zahrnout ji do ETL procesu. V některých případech lze pokračovat v loadu od místa selhání, může se ale stát, že bude nutné provést celý load znovu.

3. 6 Způsoby realizace ETL

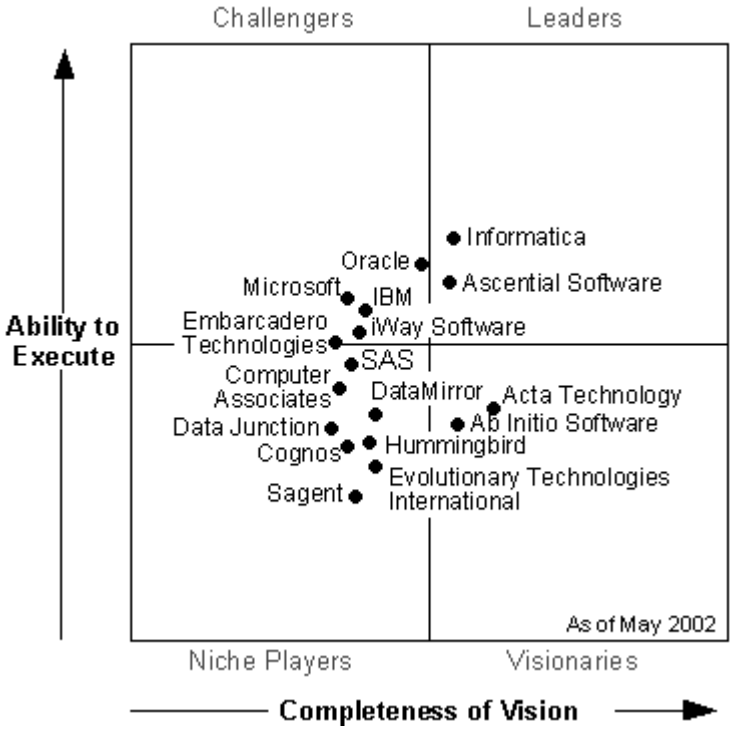
ETL je možné realizovat dvěma způsoby. Buď firma vyvine (nechá si vyvinout) vlastní ETL skripty nebo použije speciální ETL nástroj. První varianta, to znamená vývoj skriptů v SQL, jeho procedurální nadstavbě T-SQL (Microsoft) nebo PL/SQL (Oracle), C/C++, Perl a podobně, nevyžaduje vysokou počáteční investici a vývojáři jsou většinou schopni psát skripty bez nutnosti rozsáhlejšího zaškolení. Rizika tohoto řešení spočívají v náročné správě, údržbě a především v nízké transparentnosti. Při psaní dlouhého programového kódu se navíc zvyšuje pravděpodobnost výskytu chyby. Variantu vývoje vlastních skriptů lze doporučit v případech, kdy zdroje primárních dat nejsou příliš rozsáhlé a především, pokud jsou data uložena na jednotné platformě. V situaci, kdy potřebujeme přenést do datového skladu data z nehomogenních zdrojů a provádět nějaké náročnější transformace, je vhodnější použít některý z ETL nástrojů. Jejich použití přináší následující výhody [45]:

- *Vysokou produktivitu* – v grafickém prostředí je vývojář schopen navrhnout a odladit transformační proces většinou rychleji, než kdyby kód psal ručně. Také náchylnost k zanesení chyb je menší, resp. případné chyby je snadnější identifikovat a odstranit.
- *Flexibilitu* – procesy je možné modifikovat, rozšiřovat a přizpůsobovat aktuálním požadavkům.
- *Výkon* – ETL nástroje optimálně využívají HW a systémové prostředky a díky multithreadingu, paralelismu, a nativnímu přístupu k zdrojovým a cílovým systémům, dosahují maximálního výkonu.
- *Otevřenost* – vývojář se nemusí starat o to, jaký protokol či jazyk konkrétní systém používá, jen využije odpovídající komponentu.
- *Podporu metadat* – dokumentují celý systém a umožňují synchronizaci s ostatními aplikacemi datového skladu,
- *Možnost řídit a plánovaně spouštět transformace.*

3. 7 Přehled vybraných ETL nástrojů

Podle Gartner Group se trh s ETL nástroji změnil z malé skupiny dodavatelů, zaměřených téměř výlučně na ETL, na velký počet dodavatelů přicházející z rozdílných prostředí [11]:

- *„pure play" dodavatelé* - Ab Initio Software, Acta Technology, Ascential Software, Data Junction, DataMirror, Evolutionary Technologies International a Informatica
- *BI dodavatelé* - Cognos, Hummingbird, iWay Software (divize Information Builders), Sagent a SAS Institute.
- *dodavatelé databázových systémů* – „Velká trojka" – IBM, Microsoft, Oracle
- *dodavatelé ostatní infrastruktury* - Computer Associates a Embarcadero Technologies.



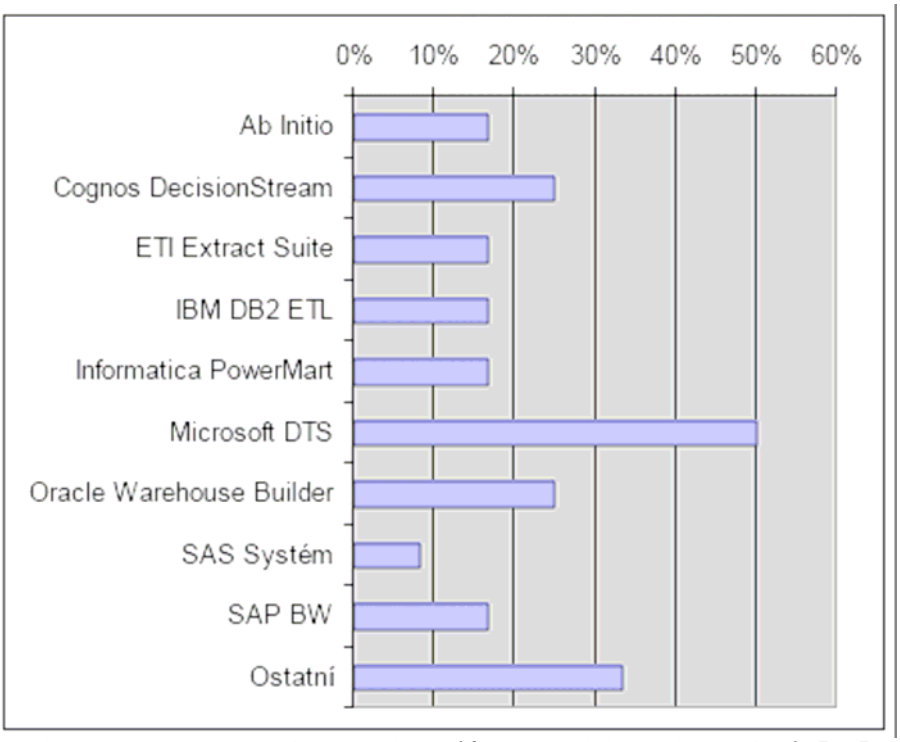
Obr. 10 - Magický ETL kvadrant. Zdroj: [11]

Výrobce	Produkt	www

Ab Initio Software	Ab Initio	http://www.abinitio.com
Cognos	Decision Stream	http://www.cognos.com
ETI	Extrakt Suite	http://www.eti.com
IBM	DB2 ETL	http://www.ibm.com
Informatica	PowerMart	http://www.informatica.com
Microsoft	DTS	http://www.microsoft.com
Microsoft	SSIS	http://www.microsoft.com
Oracle	Warehouse Builder	http://www.oracle.com
SAS	Systém	http://www.sas.com
SAP	BW ETL Tools	http://www.sap.com

Tabulka 1 - Vybrané ETL nástroje. Zdroj: [14]

Následující graf ukazuje zastoupení ETL nástrojů na českém trhu v roce 2002 – 2003 podle výzkumu [14] u 25 předních dodavatelů DW řešení, kteří tvořili podle odhadů 80% českého trhu dodavatelských subjektů.



Obr. 11 - Zastoupení ETL nástrojů na českém trhu. Zdroj: [14]

4 DATA TRANSFORMATION SERVICES

Data Transformation Services (*DTS*) je nástroj, integrovaný v Microsoft SQL Serveru 2000 pro přesouvání rozsáhlých bloků dat. Neomezuje se však pouze na kopírování, jeho pomocí lze provádět i nejrůznější transformace, používat větvení a umožňuje akce vykonávat paralelně. Umí pracovat nejenom s daty SQL Serveru[27], ale v podstatě s libovolným datovým zdrojem OLE DB[28] nebo ODBC. To z něj činí univerzální ETL nástroj.

Součástí DTS je grafické rozhraní, umožňující vývojářům snadno vytvářet komplikovaná *workflow* načítání a zpracování dat aniž by bylo nutné psát rozsáhlý programový kód. Složitější transformace lze naprogramovat v jazyce Visual Basic nebo v JavaScriptu či Perlu. Co se týče uživatelského rozhraní a logiky přístupu k transformaci, má uživatel dvě možnosti, přičemž fyzicky jde o stejný proces [25]:

- *Import/Export Data* – utilita dostupná z hlavního Windows menu MS SQL Serveru,
- *DTS Package Editor* – složka dostupná například z aplikace MS SQL Server Enterprise Manager.

4. 1 Import and Export Data

Tuto aplikaci, kterou tvoří posloupnost dialogových oken, je vhodné použít, pokud není nutné provádět složité transformace a je potřeba pouze přenést databázové tabulky, případně relační vztahy a integritní omezení. Uživatel je vyzván, aby postupně vyplnil následující údaje:

- *zdroj dat* – typ (dBase, Excel, Access, Paradox, ODBC pro Oracle, MS Visual FoxPro, textový soubor a další), parametry připojení (autorizace, výběr případného serveru a databáze nebo výběr souboru).
- *metoda pro výběr množiny údajů k přenesení* – je možné kopírovat vybrané tabulky a pohledy jako celek ze zdroje do cílové databáze, specifikovat množinu pomocí SQL dotazu nebo kopírovat objekty a údaje mezi databázemi pod správou SQL Serveru.
- *výběr zdrojových a cílových tabulek a návrh případných transformací* – u transformace je možné editovat jak SQL kód, tak transformační kód (Visual Basic nebo jiný)
- uložení, plánování a replikace dat.

Po spuštění aplikace informuje o průběhu jednotlivých etap přenosu a transformace.

4. 2 DTS Package Editor

Jak již samotný název napovídá, DTS pracuje s tzv. *balíčky* (*package*). V balíčku je možné definovat posloupnost extrakcí ze zdrojových systémů, transformací dat a načítání do cílových databází.

V DTS Editoru jsou tři základní druhy objektů, ze kterých sestavujeme ETL:

- *connection* (spojení),
- *task* (úloha),
- *workflow* (posloupnost operací).

[27] Samozřejmě nejefektivněji pracuje v rámci prostředí SQL Serveru.

[28] OLE DB (Object Linking and Embedding) – standard, který vyvinula společnost Microsoft jako reakci na kritiku ODBC, zejména kvůli nízké výkonnosti, nesystematičnosti návrhu a neobjektovému základu ODBC.

4. 2. 1 Spojení

Objekt spojení reprezentuje jak zdrojové, tak cílové systémy. Podporovány jsou:

Relační databázové systémy:

- *MS SQL Server*,
- *Oracle* (přes *ODBC*),
- lze doinstalovat *ODBC* nebo *OLE DB* ovladače např. pro *DB2*, *Informix*, *Sybase*, *Ingress* a další,
- *obecný OLE DB poskytovatel.*

Data v samostatných souborech:

- *Access,*
- *FoxPro,*
- *dBase,*
- *Excel,*
- *textové soubory,*
- *html soubor.*

4. 2. 2 Úlohy

Úloha je jednotkou práce, kterou má DTS provést. Po instalaci SQL Serveru je v Editoru připraveno velké množství úloh [52], uživateli je umožněno vytvářet i vlastní úlohy. DTS obsahují následující úlohy [52]:

- *ActiveX Script* – pomocí VB Scriptu, JScriptu nebo jiných, dodatečně nainstalovaných, skriptovacích jazyků umožňuje v balíčku provádět vlastní úlohy, které DTS neobsahuje (např. složitější transformace).
- *Analysis Services Processing* – umožňuje zpracování objektu, definovaného v SQL Serveru 2000 Analysis Services.
- *Bulk Insert (hromadné vložení)* - dokáže velmi rychle přesunout velké objemy dat, ale neumožňuje žádné transformace. Má stejnou funkci jako příkaz `BULK INSERT` nebo utilita bcp.exe.
- *Execute Package (spusť balíček)* – umožňuje opakované využití balíčků a jejich modularizaci.
- *Copy SQL Server Objects* – kopírování libovolných objektů (tabulky, views, uložené procedury, atd.) z jedné instance databáze SQL Serveru do druhé.
- *Data Driven Query (Dotaz řízený daty)* – provádí dotazy založené na jedné nebo více dynamických podmínkách.
- *Data Mining Prediction* – slouží k načtení výsledku z prediktivní úlohy dolování dat, která je integrovaná v Analysis Services. Podmínkou je, aby data miningová úloha byla umístěna na lokálním serveru.
- *Dynamic Properties* – umožňuje získání údajů z externích zdrojů, jejich uložení uvnitř balíčku a použití pro podmínečné bloky v kódu jiných součástí balíčku nebo pro dotazy řízené daty. Každá z těchto úloh může načíst pouze jednu hodnotu.
- *Execute Package (spusť balíček)* – umožňuje opakované využití balíčků a jejich modularizaci.
- *Execute process (spusť proces)* – spuštění externího programu pomocí příkazového řádku.
- *Execute SQL (spusť SQL příkaz)* – slouží k vykonání jednoho nebo více SQL nebo T-SQL příkazů.
- *FTP* – slouží ke stahování a odesílání souborů ze sídel, resp. na sídla FTP.

- *Message Queue* – pokud je nainstalován MS Message Queue, přidá zprávu do fronty zpráv.
- *Send Mail* – pošle mail jednomu či více příjemcům.
- *Transfer Databases* – přesun databází mezi instancemi SQL Serveru (instance nemusí být stejné verze).
- *Transfer Error Messages* – přenese systémové zprávy mezi instancemi serverů.
- *Transfer Jobs and Logins* – přenos jobů a uživatelských profilů mezi instancemi SQL Serveru.
- *Transfer Master Stored Procedures* – umožňuje přenos uložených procedur mezi databázemi *master* různých instancí SQL Serveru.
- *Transform Data* – používá se k přesunu dat, během něhož lze data transformovat.

4. 2. 3 Posloupnost operací

Pomocí posloupnosti operací lze řídit celý ETL proces. Určuje jaké operace, za jakých podmínek a v jakém pořadí, se mají vykonat. Je vytvořena vazba mezi dvěmi úlohami nebo celými balíčky. K dispozici jsou tyto volby [52]:

- *On Completion (po dokončení)* – druhá (závislá) úloha je spuštěna po dokončení první. Nezáleží na tom, zda první úloha proběhla úspěšně, či nikoli.
- *On Success (při úspěchu)* – závislá úloha se spustí jen při úspěšném provedení první úlohy.
- *On Failure (při selhání)* – spustí se v případě, že první úloha z libovolného důvodu neproběhla celá nebo všechny její operace nebyly úspěšné. Je více než vhodné používat volby On Success a On Failure dohromady.

4. 3 Použití DTS v praxi

V této části bych rád ukázal použití DTS na reálném příkladě ETL řešení. Projekt pro Generální ředitelství cel ČR realizovala společnost Adastra, s. r. o.

4. 3. 1 Profil společnosti Adastra, s. r. o.

Adastra je mezinárodní, technologicky nezávislá konzultační společnost, která dodává softwarová řešení a služby. Specializuje se zejména na Business Intelligence, což zahrnuje DW, *MIS* (*Management Information System*), *CRM*, data mining, *MDM* (*Master Data Management*), atd. Další kompetencí je aplikační vývoj, na platformách Microsoft.NET Framework a J2EE. Adastra používá sofistikované projektové metodiky, které jsou zárukou úspěšného dokončení realizovaných projektů. Vedle projektových metodik založených na mezinárodních metodologiích a standardech (např. PMI-PM BOK, PRINCE2, ITIL) mezi ně patří i řada vlastních metodik, například pro analýzu business požadavků (Business Discovery), metodika integrovaného CRM (CRM.360) nebo metodika implementace datových skladů (DW.360) a další [1]. Od roku 2000 je držitelem certifikátu ISO 9001 od mezinárodní agentury Moody International. Evropské aktivity firmy jsou řízeny z centrály v Praze a prostřednictvím poboček v Bratislavě, Ostravě a Frankfurtu n. M. Severoamerická kancelář Adastry je v kanadském Torontu. Adastra má přes 350 zaměstnanců, z toho 250 v ČR a SR. Roční obrat překračuje 500 mil. Kč a roční růst je více než 40%. Mezi zákazníky patří telekomunikační operátoři, bankovní a finanční instituce, obchodní řetězce a subjekty státní správy. Za projekty pro Ahold Czech Republic, Allianz pojišťovnu, Bank of Montreal, Citibank, Českou pojišťovnu, Českou spořitelnu, ČSOB, DHL, Eurotel, HVB Bank, Kooperativu, Komerční banku, Ministerstvo obrany ČR, Telefónicu O2, Vodafone, Zentivu a další získala řadu ocenění, mezi něž patří např: Top 10 IT firma roku 2006, Top 10 systémový integrátor roku 2004 a 2006, Canada's 50 Best Manager Companies 2006, Fast 50 Central Europe 2005 a progresivní zaměstnavatel roku 2006 od agentury Czech Invest.

4. 3. 2 Profil zákazníka

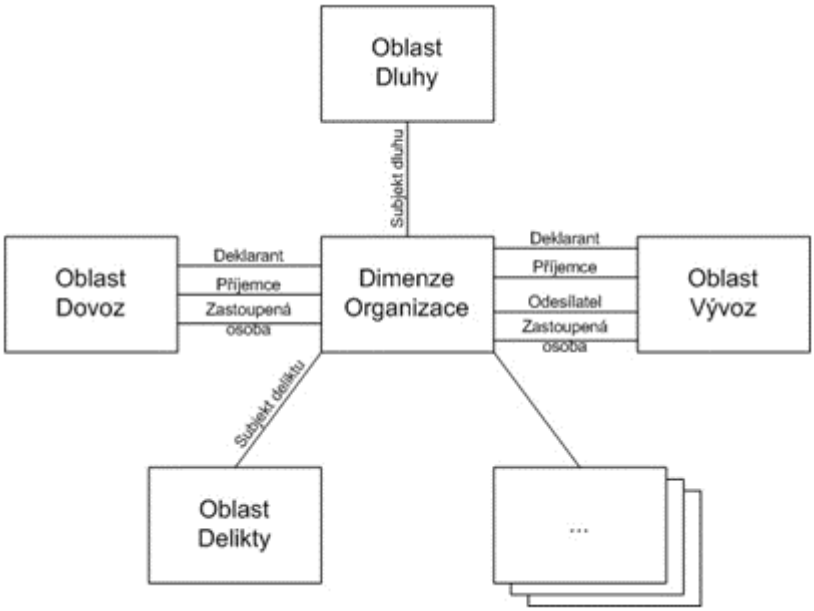
Celní správu České republiky (CS ČR) tvoří Generální ředitelství cel s pozicí správního úřadu s celostátní působností, dále osm celních ředitelství a jim podřízených 54 celních úřadů s vymezenou územní působností. Celní správa je bezpečnostním sborem a její činnost zapadá do systému celního dohledu nad zbožím v rámci jednotného celního území Evropské unie. Celní správa České republiky je také výhradním správcem spotřebních daní, byla jí svěřena kontrolní oprávnění v oblasti nákladní silniční dopravy, kontrola v oblastech zahraničního obchodu s vojenským materiálem a provádění společné zemědělské politiky Evropského společenství. Další oblastí činnosti CS ČR je kontrola nakládání s odpady, obchodu s chráněnými druhy fauny a flóry a nelegální zaměstnanosti cizinců. V neposlední řadě je CS ČR zařazena do integrovaného záchranného systému země [42].

4. 3. 3 Popis projektu

Před nasazením DW byla data rozmístěna v mnoha primárních systémech, např. v mzdovém, účetním, personálním, fakturačním a k tomu dále v dalších specializovaných systémech, řešící specifickou problematiku celní správy [42]. Vzhledem k tomu, že tyto systémy jsou od různých dodavatelů a využívají různé technologie, každý systém obsahoval vlastní číselníky. Navíc odkazy na doklady nebo dokumenty jiných systémů byly realizovány zápisem volného textu bez ověřování, takže docházelo k problémům s referenční integritou. Provádění konsolidovaných datových výstupů proto bylo obtížné, problémové a někdy dokonce nemožné. Dotazy trvaly neúnosně dlouhou dobu a kladly velké nároky na jejich tvůrce.

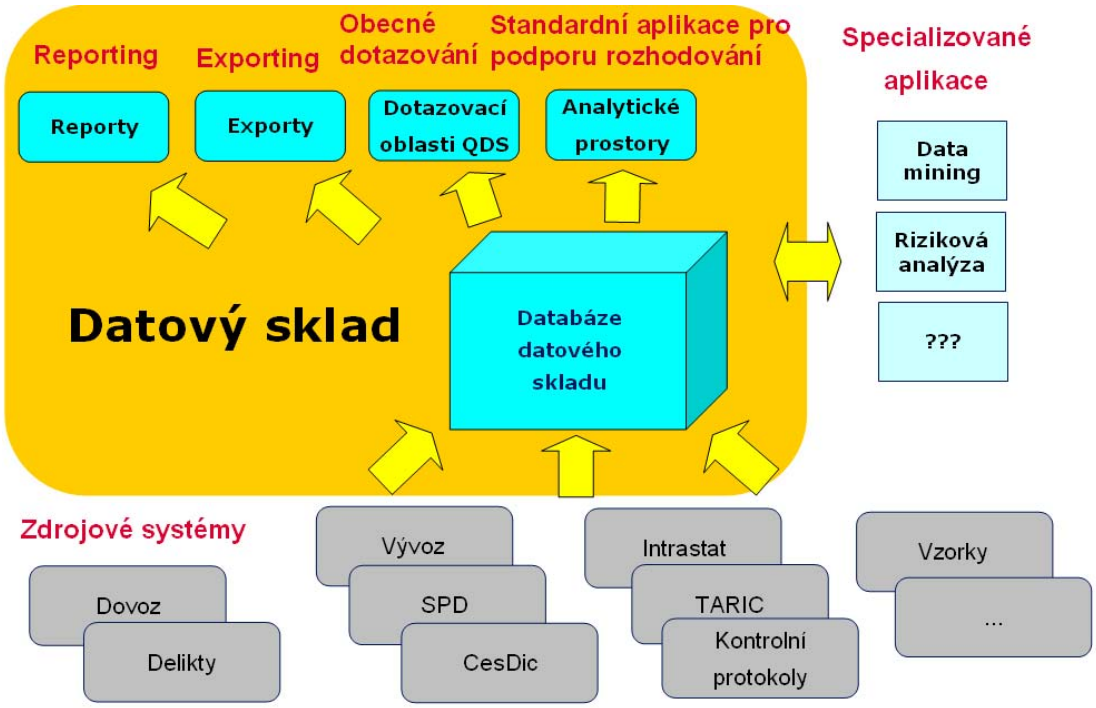
Účelem řešení konsolidovaného DW, který musí obsahovat data integrovaná, vyčištěná a validovaná (včetně historických), bylo zajistit [42]:

- provádění *on-line analýzy dat, ad-hoc dotazování* s možností velmi rychlé odezvy,
- možnost *pokládání „mlhavých" dotazů* s jejich postupným zpřesňováním, pohledy na data přes jednotlivé agendy, nezávislost na tvůrcích reportů a to vše z jednotné datové základny,
- snadnou *dostupnost dat,*
- *jednotnou verzi pravdy*,
- *odlehčení primárním systémům*, protože před implementací DW skladu zajišťovaly i úlohu reportingu a exportingu.



Obr. 12 - Konsolidace oblastí/agend/evidencí. Zdroj: [53]

Implementované řešení DW současně zastává funkci datové základny pro další specializované aplikace CS ČR a je součástí projektů, které byly realizovány v rámci programu PHARE.
V současné době se systémem pracuje přes 400 uživatelů, ale vzhledem k tomu, že k DW má přístup několik tisíc zaměstnanců CS ČR, počítá se s dalším nárůstem počtu uživatelů [42].



Obr. 13 - Základní komponenty DW CS ČR. Zdroj: [53]

4. 3. 4 Použité technologie

Relační vrstva:
- databázový stroj MS SQL Server 2000 Enterprise Edition 64 bit

ETL:
- MS SQL Server 2000 DTS, Job Agent, T-SQL

Multidimenzionální vrstva:
- na straně serveru:
  - MS SQL Server 2000 Analysis Services
- na straně klienta:
  - MS Excel Add-in for SQL Server Analysis Services
  - ProClarity

Reporty:
- reportovací server:
  - MS SQL Server 2000 Reporting Services
- klient:
  - Internet Explorer

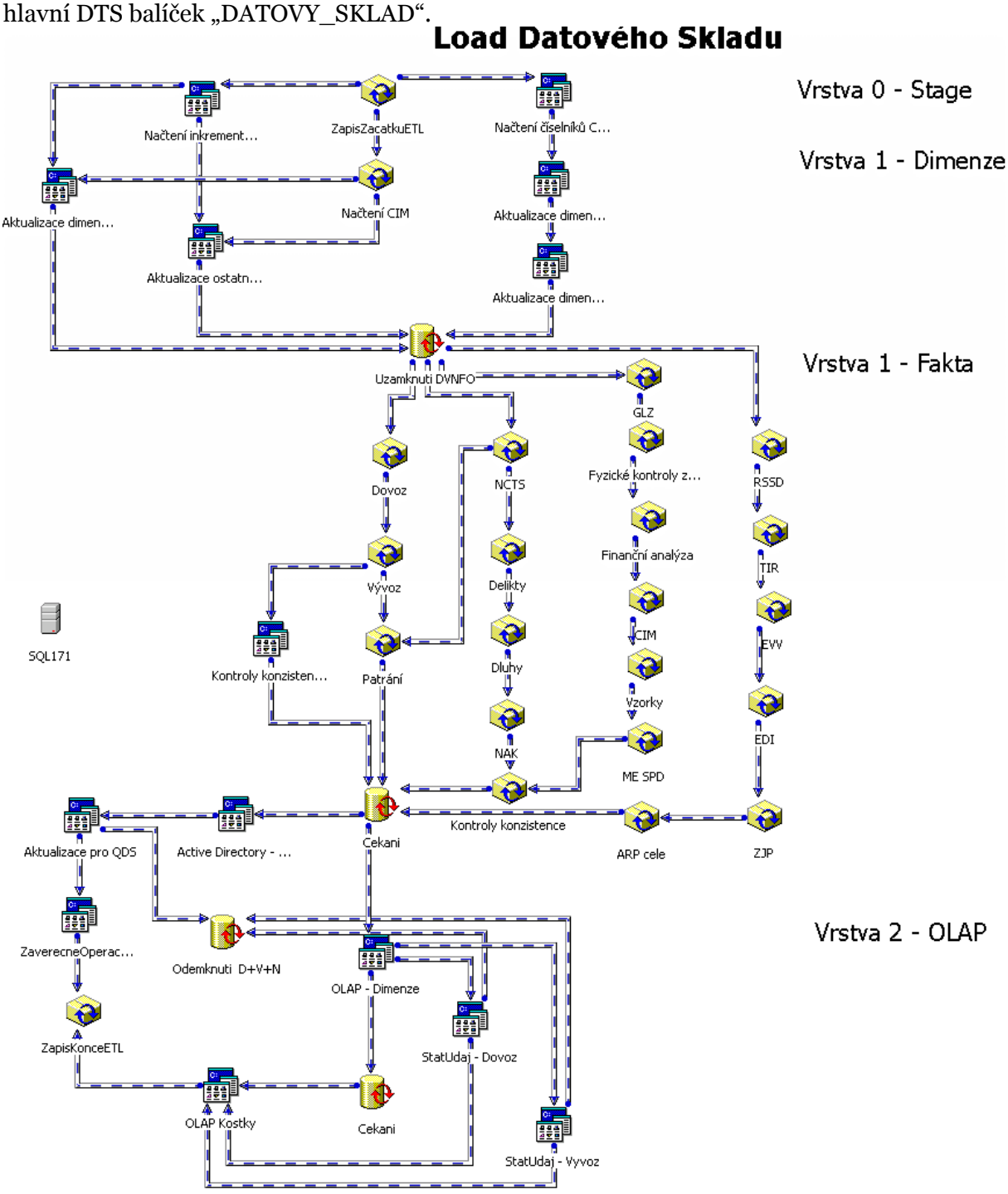
Exporty:
- bcp utilita
- uživatelské rozhraní:
  - Reporting Services, Internet Explorer

Obecný dotazovací nástroj:

- QDS (na zakázku vyrobený systém) [53]. V rámci tohoto nástroje uživatel vytvoří a spustí dotaz do datového skladu. Výsledkem dotazu jsou záznamy vyhovující zadaným kritériím.

4. 3. 5 ETL vrstva

Jak jsem již uvedl, ETL bylo v projektu realizováno pomocí DTS. Následující obrázek ukazuje hlavní DTS balíček „DATOVY_SKLAD“.



Obr. 14 - Hlavní DTS balíček “DATOVY_SKLAD”.

ETL proces je rozdělen na čtyři části. V první části jsou data načítána do Data Staging Area (Stage), v druhé jsou v datovém skladu aktualizovány dimenze, třetí část zajišťuje načtení dat do faktových tabulek a čtvrtá má na starost aktualizaci OLAPu. Není mým cílem celý ETL proces detailně popsat, protože bych značně překročil rámec této práce, a to jak z hlediska obsahu, tak z hlediska rozsahu. Rád bych zde nastínil způsob provedení ETL a detailnějšímu rozboru podrobím balíček „IntrastatLoad“, který je spouštěn samostatně a obsahuje specifický ETL proces pro data Intrastatu[29]. Nejprve ukáži stávající řešení, vybudované v DTS a v závěrečné části se tuto část ETL procesu pokusím realizovat v SSIS.

Prvním balíčkem je „ZapisZacatkuETL“, který využívá proceduru zapisující údaje o loadu do speciální auditovací tabulky „TSEtlLog“. Zaznamenávají se zde chyby, důležité události, pomocí kterých lze rekonstruovat podmínky existující v datovém skladu v určité době, a detaily o ETL procesu. Tyto informace jsou důležité při hledání příčin jevů, které v DW nastaly a při hledání důvodu doby trvání procesu a možností jeho krácení. Tabulka „TSEtlLog“ obsahuje sloupce s následujícím významem:

Sloupec	Popis
`EtlLogId`	Umělý primární klíč
`TypZaznamu`	Identifikuje, zda se jedná o chybu, nebo informaci o trasování
`Priorita`	Info, Warning, Error, System error
`NazevProcesu`	Název procesu, kde se událost stala
`Cas`	Čas, kdy k události došlo
`Popis`	Popis události
`Komentar`	Detailní popis události (nepovinný)
`Uroven`	Požadovaná úroveň logování
`Uzivatel`	Aktuální uživatel
`EtlProcesId`	Odkaz na ETL proces, který záznam generoval

Tabulka 2 - Obsah tabulky "TSEtlLog"

Každá tabulka v datovém skladu obsahuje sloupce „VlozilEtlProcesId“ a „AktualizovalEtlProcesId“, které zajišťují vazbu auditu na data.

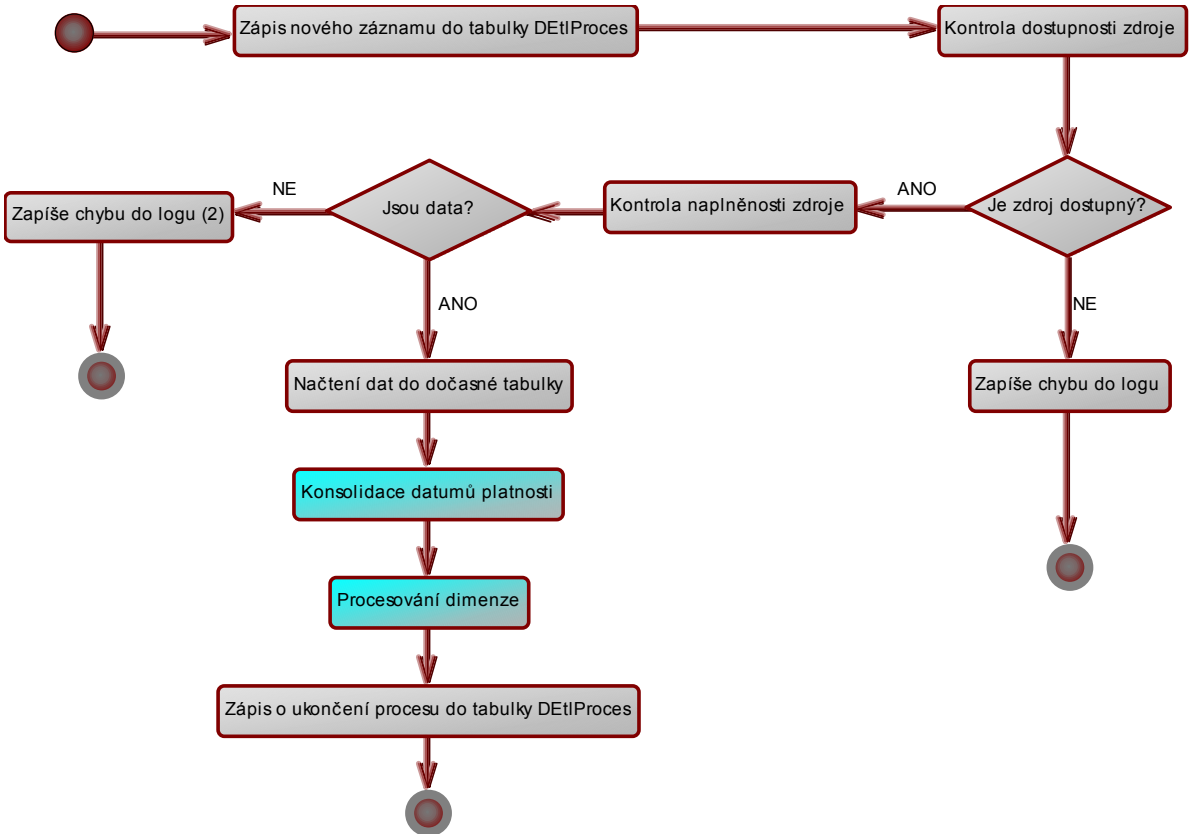
V dalším kroku jsou paralelně spuštěny tři procesy z nichž první dva mají za úkol naplnit nultou vrstvu DW neboli Stage, kterých je zde celkem zhruba 20. Jeden balíček zajišťuje load dat uložených na SQL Serveru, kde jsou všechny procesy načtení volány sériově, protože vzhledem k tomu, že předpokládaným úzkým místem by byla síťová komunikace, paralelizace by nespíš nepřinesla zvýšení efektivity. Síťová komunikace je bezesporu limitujícím prvkem, ale vzhledem k načítání pouze inkrementů dat rozsáhlých zdrojových systémů se po síti přenáší relativně nízké objemy dat. Load do Stage vrstvy probíhá ve 4 současně běžících procesech, aby v případě uplatnění maximální paralelizace SQL Serverem měly možnost tyto procesy běžet vedle sebe. V praxi běží najednou většinou dva až tři.

Dále jsou do Stage načtena zdrojová data z databáze IBM Informix. Nejprve je vytvořen její odlitek, poté jsou data exportována do textového souboru (soubor .unl) a po úpravě ukončení řádků, tak aby bylo kompatibilní s SQL Serverem (standardní CR-LF), je pomocí uložené procedury naplněna odpovídající Stage. Třetí proces má za úkol načíst zdrojové číselníky, které jsou uloženy v databázi IBM DB2. Ta ale není přístupná přímo přes ODBC rozhraní, ale jsou načítána exportovaná data ve formátu .dbf.

Po provedení první části ETL procesu, který naplní nultou vrstvu, dojde k aktualizaci vrstvy 1, to znamená dimenzionálních a faktových tabulek. Dimenze jsou aktualizovány pomocí úloh

[29] Intrastat je systém sběru dat pro statistiku obchodu se zbožím mezi členskými státy Evropské unie, pokud při jeho přijetí nebo odeslání není povinnost předkládat celním orgánům celní prohlášení. Vykazují se v něm údaje o vnitrounijním obchodu.

volajících uložené procedury. Aktualizace některých, zvláště komplikovaných, oblastí je prováděna zvlášť. Mezi dimenzemi je třeba rozlišovat typy *SCD0*, *SCD1 a SCD2*[30]. Jejich zpracování je rozdílné, obecný proces načítání dimenzí je znázorněn na následujícím diagramu.



Obr. 15 - Obecný proces ETL dimenze

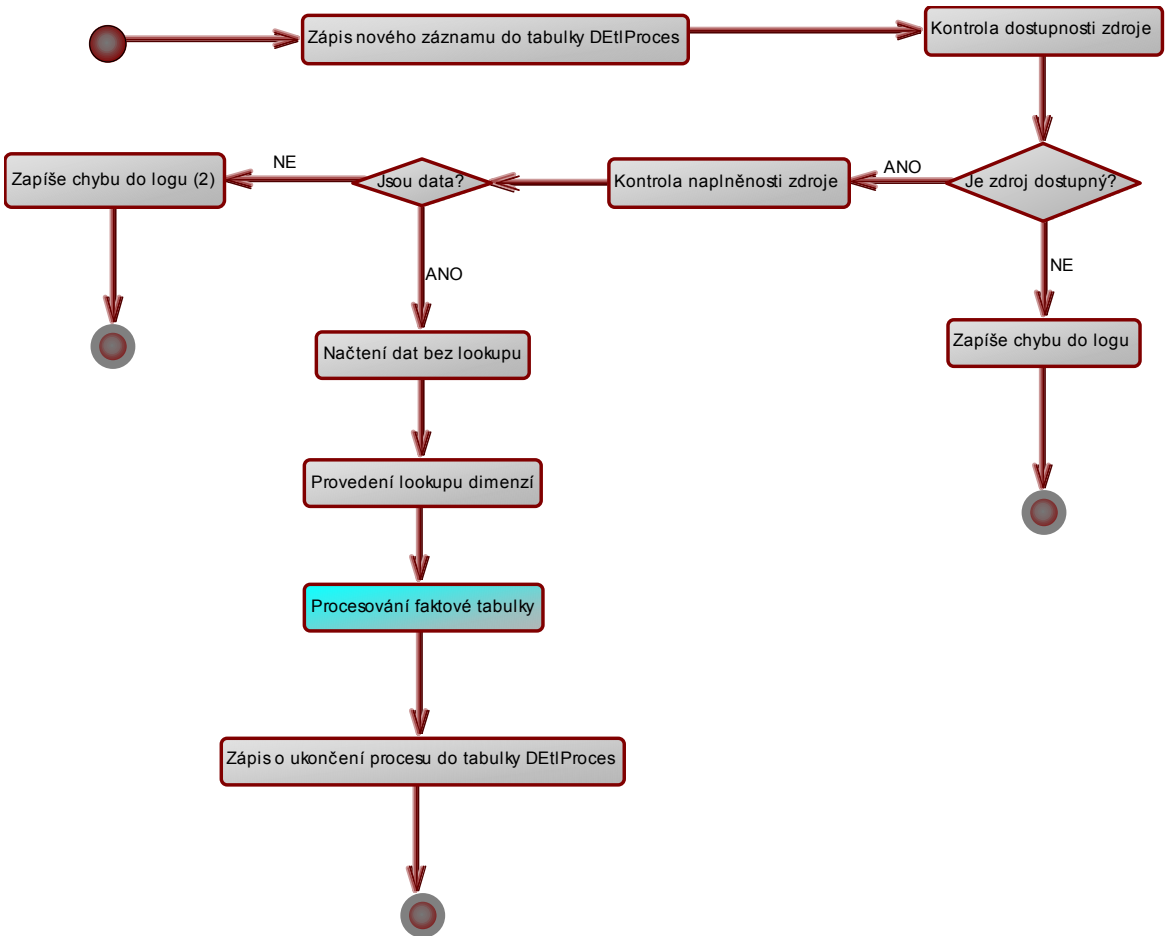
Pokud je zdroj dostupný a jsou v něm data, dojde k vytvoření obrazu dimenze ze zdrojových dat, v opačném případě je generována chyba do logu a proces je ukončen. Obraz dimenze musí mít, až na technické sloupce, totožnou strukturu. Je naplněn zpravidla všemi daty a předán obecné proceduře, která identifikuje změněné, nové a vymazané záznamy. Poté provede aktualizaci změněných záznamů, vložení nových a u vymazaných nastaví neplatný interval platnosti podle zdrojového systému[31], pokud se platnost sleduje. Časové obory platnosti, přebírané ze zdrojových systémů, a obory platnosti datového skladu je třeba konsolidovat. Dále jsou doplněny technické hodnoty „`#neuvedeno`“ (nahradí `NULL`, nulu, prázdný řetězec a mezery na vstupu) a „`#neznámo`“ (není-li nalezen odkaz na dimenzi).

ETL faktů probíhá podobně, ale nahrazuje mnohé atributy na vstupu odkazy do dimenzí. Toto nahrazování nazýváme *lookupem*. V první fázi se provádí lookup s ohledem na referenční datum, ve druhé jsou pro nenalezené instance hledány poslední výskyty kódu v dimenzi bez ohledu na datum a nakonec se zbylé kódy, nedohledané v dimenzi, do této dimenze přidají. Aby mohla být provedena třetí fáze lookupu, musí dimenze obsahovat

[30] SCD0 – nedochází k aktualizaci starých záznamů, jen se přidávají nové. Všechny atributy dimenze jsou zpravidla součástí přirozeného klíče. SCD1 – nová hodnota přepíše stávající, aktualizují se změněné hodnoty, mohou se i historizovat. SCD2 – údaje se nepřepisují, přidají se nové a u původních se nastaví příznak vypršení platnosti. Dalším typem je SCD3, takové dimenze se ale na projektu nevyskytují.

[31] Záznamy nejsou z dimenzí nikdy mazány, protože na ně může existovat odkaz z faktové tabulky.

atribut „`#neznámo`“. Následující diagram ukazuje obecný postup zpracování faktových tabulek.



Obr. 16 - Obecný proces ETL faktů

Zpracování faktů probíhá ve čtyřech proudech podle procesorů. Proudy jsou vyváženy podle času, takže vždy běží minimálně čtyři úlohy a proudy by měly skončit přibližně ve stejnou dobu. Po aktualizaci fakt je datový sklad pomocí uložené procedury po oblastech uzamknut a jsou provedeny kontroly konzistence. Probíhají zvlášť pro komplikované oblasti, obsahující data týkající se vývozu a dovozu, protože trvají nejdéle, poté jsou provedeny ostatní kontroly konzistence. Kontrola je realizována pomocí dvou dotazů – do zdrojového systému a datového skladu. Datasety jsou pak spárovány, výsledek je zaznamenán do tabulky „TFVysledekKontroly“ a případný nesoulad generuje chybu do logu.

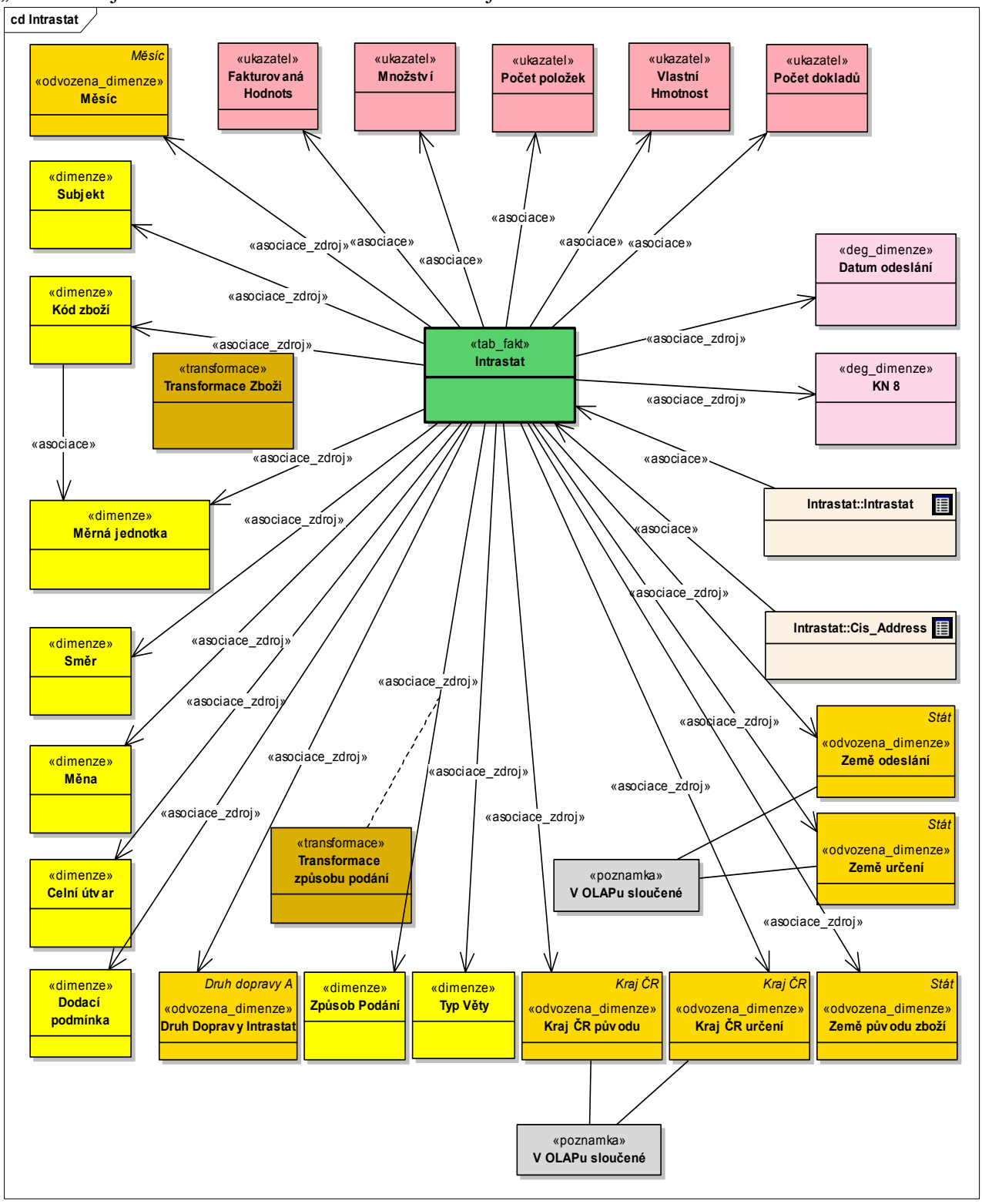
Čtvrtá část ETL, zpracování vrstvy 2 (OLAP), je zahájena po odemknutí pro dotazovací aplikace nad datovým skladem a zápisu do logu. OLAP dimenze jsou zpracovávány inkrementálně a protože přepočítání agregací trvá téměř dvě hodiny, je před přeprocesováním OLAP kostek nastaveno čekání 1h 50 min. Tento čas je využit pro načítání dalších pomocných struktur. Nejdéle trvá přepočet statistických údajů pro zmiňované komplikované oblasti. Čas je využit k načtení a zpracování údajů z Active Directory[32] a

[32] Active Directory slouží k ověřování uživatelů. Uživatelé, kteří jsou zařazeni v doméně nemusí při komunikaci s datovým skladem zadávat uživatelské jméno a heslo, ověření uživatele proběhne pomocí active directory. To se týká veškerých nástrojů datového skladu. Pokud uživatel není zařazený v doméně, tak při komunikaci pomocí QDS a reportovacího nástroje Reporting Services bude dotázán na přístupové jméno a heslo.

aktualizaci plošných struktur pro QDS. Po dokončení těchto operací je zapsáno do logu o provedení a ukončení ETL procesu. Průměrná doba načtení denního přírustku se pohybuje od 4 do 5 hodin pro relační vrstvu datového skladu a cca 4 hodiny pro multidimenzionální vrstvu (OLAP).

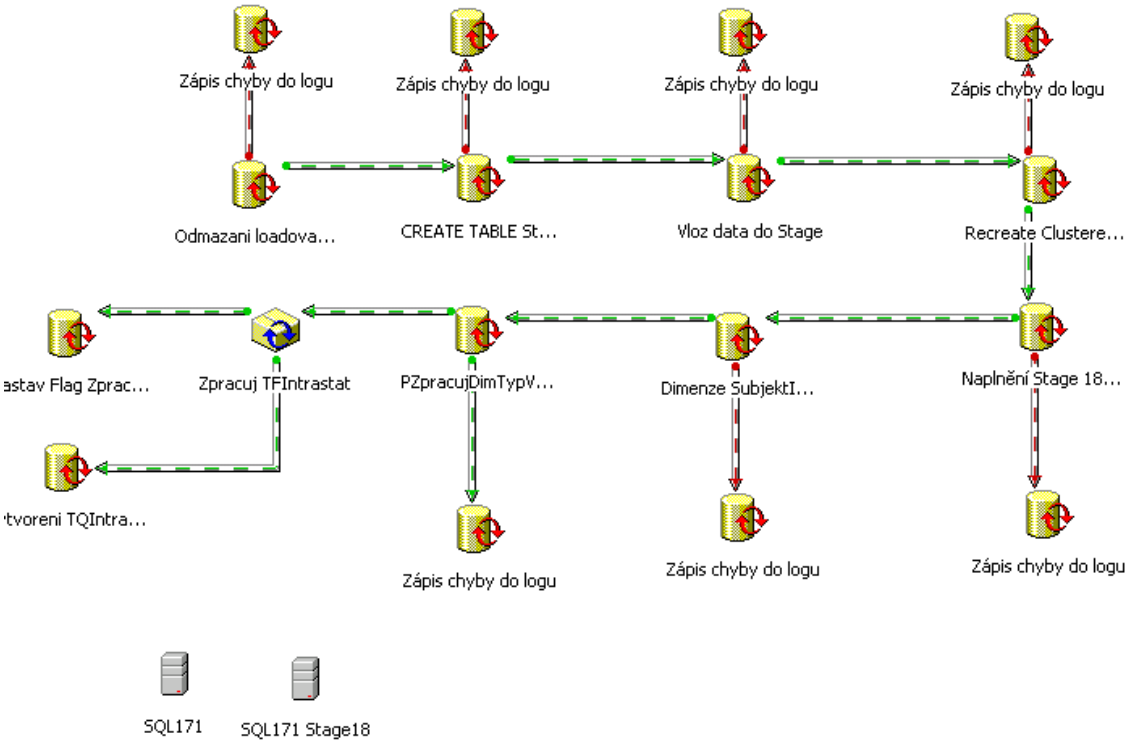
4. 3. 6 ETL pro oblast Intrastat

„Intrastat“ je samostatné datové tržiště s následující strukturou.



Obr. 17 - Struktura datového tržiště "Intrastat"

Zdrojem pro tuto část DW je databáze Intrastatu získávaná z Českého statistického úřadu na přenosných médiích. Data jsou poté přenesena do prostředí MS SQL Serveru 2000. Celní správa databázi obdrží jednou měsíčně a data v ní mají dva měsíce zpoždění (např. v průběhu května jsou získána data za březen). Z toho plyne, že ETL proces pro datové tržiště „Intrastat“ musí probíhat samostatně a v jiném intervalu než pro ostatní části datového skladu. Je spouštěno jednou za měsíc a zajišťuje ho balíček „IntrastatLoad“.



Obr. 18 - Balíček “IntrastatLoad”

Ze způsobu zpracování dat vyplývá, že v oblasti „Intrastat“ nebudou sledovány změny ve faktových datech. Pokud tedy dojde ke změně ve faktových datech, nebude tato změna identifikována a dojde k přepisu původních dat novými daty. Původní data nebudou dohledatelná. Nejprve je tedy spuštěn balíček, obsahující SQL příkazy, které odmažou vybraná data z faktové tabulky “TFIntrastat”, uložené v datovém skladu. Jestliže budou ve zdrojové databázi data starší než půl roku, vymažou se data za aktuální rok (od 1. ledna). V opačném případě jsou vymazána data od prvního ledna předcházejícího roku.

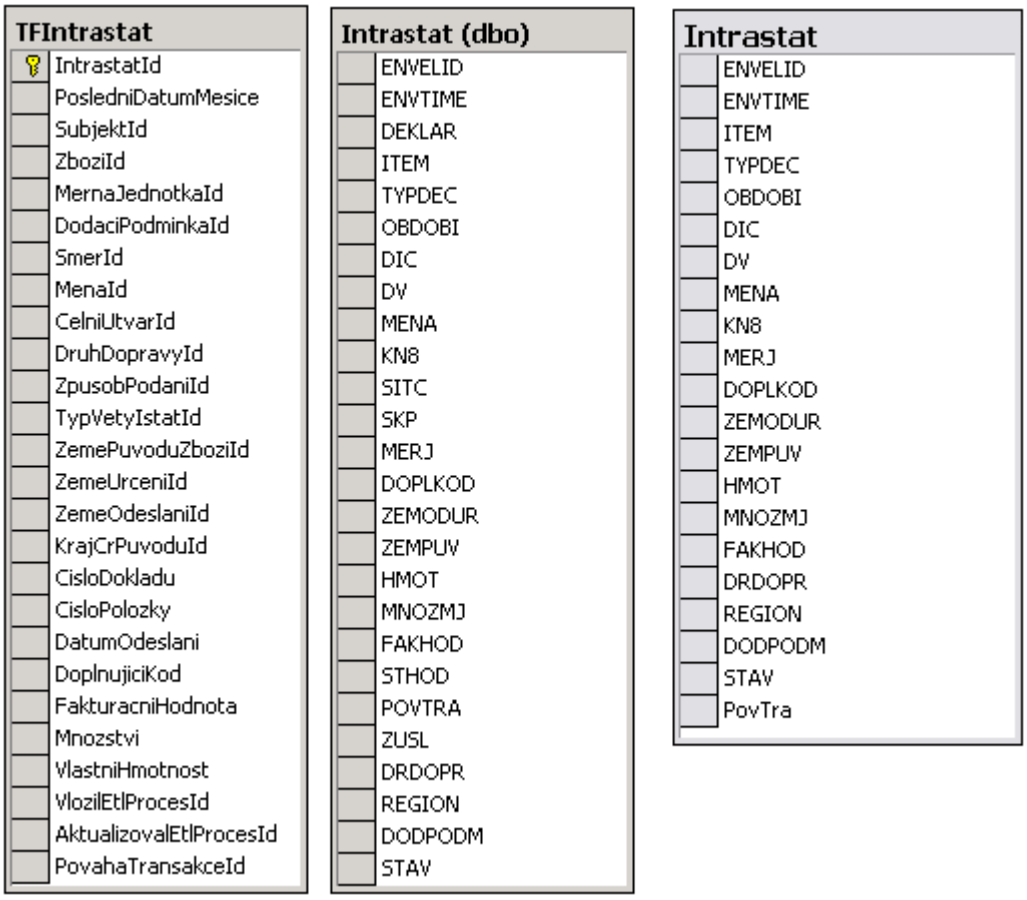
```
DECLARE @LastPeriod varchar(6), @LastMonth smallint, @StartingPeriod as varchar(8), @Cmd
varchar(8000)

SELECT @LastPeriod = MaxObdobi FROM OPENQUERY (dszdroj1, 'SELECT MAX(Obdobi)  MaxObdobi FROM
Istat.dbo.Intrastat ')

SET @LastMonth = CAST(RIGHT(@LastPeriod,2) as smallint)

if @LastMonth > 7
        SET @StartingPeriod = LEFT(@LastPeriod,4) + '0101'
else
        SET @StartingPeriod = CAST((CAST(LEFT(@LastPeriod,4) as int)-1) as varchar(6)) +
'0101'

DELETE FROM DS.dbo.TFIntrastat WHERE PosledniDatumMesice >= @StartingPeriod
```



Obr. 19 – Faktová tabulka TFIntrastat, zdrojová a Stage tabulka Intrastat

Další balíček znovu vytvoří tabulku “Intrastat” ve Stage. Ta má stejnou strukturu jako zdrojová, ale neobsahuje některé nepotřebné sloupce (jak ukazuje Obr. 19). Obsahuje hlášení o Intrastatu. Řádek tabulky obsahuje řádek z hlášení. Údaje společné pro všechny evidence jsou nakopírovány na každý řádek hlášení.

```
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[Intrastat]') and
OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[Intrastat]
GO

CREATE TABLE Stage18.[dbo].[Intrastat] (
          [ENVELID] [nvarchar] (50) COLLATE Czech_CI_AS NULL ,
          [ENVTIME] [datetime] NULL ,
          [ITEM] [nvarchar] (50) COLLATE Czech_CI_AS NULL ,
          [TYPDEC] [nvarchar] (2) COLLATE Czech_CI_AS NULL ,
          [OBDOBI] [nvarchar] (6) COLLATE Czech_CI_AS NULL ,
          [DIC] [nvarchar] (12) COLLATE Czech_CI_AS NULL ,
          [DV] [nvarchar] (1) COLLATE Czech_CI_AS NULL ,
          [MENA] [nvarchar] (3) COLLATE Czech_CI_AS NULL ,
          [KN8] [nvarchar] (8) COLLATE Czech_CI_AS NULL ,
          [MERJ] [nvarchar] (3) COLLATE Czech_CI_AS NULL ,
          [DOPLKOD] [nvarchar] (12) COLLATE Czech_CI_AS NULL ,
          [ZEMODUR] [nvarchar] (3) COLLATE Czech_CI_AS NULL ,
          [ZEMPUV] [nvarchar] (3) COLLATE Czech_CI_AS NULL ,
          [HMOT] [numeric](14, 0) NULL ,
          [MNOZMJ] [numeric](14, 0) NULL ,
          [FAKHOD] [numeric](14, 0) NULL ,
          [DRDOPR] [nvarchar] (1) COLLATE Czech_CI_AS NULL ,
          [REGION] [nvarchar] (2) COLLATE Czech_CI_AS NULL ,
          [DODPODM] [nvarchar] (3) COLLATE Czech_CI_AS NULL ,
          [STAV] [nvarchar] (1) COLLATE Czech_CI_AS NULL ,
          [PovTra] [varchar] (2) COLLATE Czech_CI_AS NULL
) ON [PRIMARY]
```

Poté jsou do této tabulky načtena data. Logika výběru dat je podobná té v prvním balíčku.

```
DECLARE @LastPeriod varchar(6), @LastMonth smallint, @StartingPeriod as varchar(8), @Cmd
varchar(8000)

SELECT @LastPeriod = MaxObdobi FROM OPENQUERY (dszdroj1, 'SELECT MAX(Obdobi) MaxObdobi FROM
Istat.dbo.Intrastat ')

SET @LastMonth = CAST(RIGHT(@LastPeriod,2) as smallint)

if @LastMonth > 7
        SET @StartingPeriod = LEFT(@LastPeriod,4) + '01'
else
        SET @StartingPeriod = CAST((CAST(LEFT(@LastPeriod,4) as int)-1) as varchar(6)) + '01'

SET @CMD =
'insert into stage18..intrastat (EnvelId, ENVTIME, item, TYPDEC, OBDOBI, DIC, DV, MENA, KN8,
MERJ, DOPLKOD, ZEMODUR, ZEMPUV, HMOT, MNOZMJ, FAKHOD, DRDOPR, REGION, DODPODM, Stav, POVTRA )
select * from openquery(dszdroj1,''select EnvelId, ENVTIME, Deklar, TYPDEC, OBDOBI, DIC, DV,
MENA, KN8, MERJ, DOPLKOD, ZEMODUR, ZEMPUV, HMOT, MNOZMJ, FAKHOD, DRDOPR, REGION, DODPODM,
Stav, POVTRA from Istat.dbo.Intrastat WHERE Obdobi >= ''''' + @StartingPeriod + ''''' ORDER BY
EnvelId, Deklar'')'

EXEC (@Cmd)
```

Během procesu načítání je vypnuta reindexace, která by jinak probíhala po vložení každého nového řádku. Načítání tak probíhá velmi rychle, kolem 20 milionů řádků se načítá 3 až 5 minut. Po načtení dat jsou znovuvytvořeny indexy.

```
CREATE CLUSTERED INDEX IX_Istat_EnvelIdItemObdobi ON [dbo].[Intrastat]([ENVELID], [ITEM],
[OBDOBI])
WITH  STATISTICS_NORECOMPUTE  ON [PRIMARY]
```

V další části ETL procesu jsou načtena data číselníků do Stageových tabulek “Cis_Address”, “Cis_Zeme”, “Cis_Trans” a “Cis_Kodpohybu”. Zdrojem jsou tabulky shodného názvu, uložené ve stejné databázi jako tabulka “Intrastat”. Načítání dat je realizováno pomocí uložených procedur. Procedury provádějí následující kontroly:

- *kontrola dostupnosti zdroje,*
- *kontrola, zda tabulka Intrastat obsahuje data,*
- *kontrola, zda každá zdrojová tabulka obsahuje data.*

Údaje o průběhu přenosu, jako jsou např. počet vložených záznamů pro každou tabulku, datum zdrojových dat, datum předchozího loadu a poslední naloadované období Intrastat, jsou zapisovány do tabulky “TSEtlLog”.

Další dva balíčky mají za úkol aktualizaci dimenzí. První z nich aktualizuje dimenzi “TDSubjekt”. Nejprve jsou pomocí procedury “PZpracujDimSubjekt” načtena data z tabulky “Cis_Address” ze Stage, uložena do dočasných tabulek a identifikovány a odstraněny duplicity ve zdroji. Dále je na správný formát transformováno DIČ. Poté je dočasná tabulka se zpracovanými daty předána proceduře “PZpracujDimSubjektPrirustek”, která zjistí, zda vkládaný záznam v tabulce “TDSubjekt” již není. Jestliže ano, porovná jméno a adresu a pokud se liší, upraví záznam. U totožných záznamů neprovede žádnou akci a neexistující vloží jako nové.

Po úspěšném dokončení aktualizace dimenze “TDSubjekt” je spuštěn balíček pro zpracování dimenze “TDTypVetyIstat”. Zpracování je zajištěno uloženou procedurou “PZpracujDimTypVetyIstat“, která po nezbytných kontrolách spustí obecnou proceduru „PProcesujDimSCDo“. Ta do dimenze vloží nové záznamy ze Stageové tabulky „Cis_Kodpohybu“ a validuje je pomocí uživatelsky definovaných kontrol, což je porovnání dat v DW s nastavenými vlastnostmi dat. Tímto způsobem lze identifikovat duplicity, nevyplněné

a neznámé hodnoty a/nebo provést kontrolu vůči masce. Každá nová kombinace podezřelých hodnot je zapsána do dimenze „TDPodezreni“ a faktové tabulky „TFPodezrelyZaznam“. Podezřelá hodnota v kontrolované tabulce pak obsahuje odkaz na dimenzi ve sloupci „PodezreniId“.

Aktualizace ostatních dimenzí, pocházejících z číselníků jiné zdrojové databáze, se provádí samostatně a v jiném časovém intervalu (každý den), v rámci hlavního DTS balíčku „DATOVY_SKLAD“.

Faktová data jsou zpracovávána pomocí „Zpracuj TFIntrastat“, což je balíček typu Execute Package Task.

Obr. 20 - Balíček “Zpracuj TFIntrastat”

Vlastní manipulace s daty je provedena uloženou procedurou “PZpracujFaktaIntrastat”. Ta vloží data ze Stageové tabulky “Intrastat” do dočasných tabulek, poté provede lookup na časovou dimenzi (místo datumů jsou vloženy odkazy na dimenzi), lookup na ostatní dimenze a zavolá obecnou proceduru “PProcesujFaktaInkrement”, která identifikuje a vloží nové záznamy a upraví změněné záznamy v souvisejících dimenzích. Poté provede uživatelsky definované kontroly a vloží upravená data do tabulky “TFIntrastat”.

Objem dat, který je potřeba načíst do faktové tabulky je poměrně velký. Měsíční dávka totiž obsahuje do šestého měsíce (včetně) data za aktuální a předchozí rok a od sedmého měsíce pak data jen za aktuální rok. Zpracování faktové tabulky proto trvá 6 – 10 hodin. Měsíční přírustek obsahuje zhruba 1 milion vět (existuje přibližně 20 tisíc deklarantů, podávajících měsíčně výkaz, který obsahuje více řádků). Změny v rychlosti nárůstu dat se nepředpokládají. Celkově “TFIntrastat” obsahuje 25 milionů záznamů.

Po úspěšné aktualizaci faktové tabulky jsou spuštěny poslední dva balíčky. Jeden má na starost vytvoření plošných struktur pro dotazování v QDS (“Vytvoření TQIntrastat”). Je vytvořena tabulka “TQIntrastat”, jejíž struktura vychází z faktové tabulky “TFIntrastat”. Má ale menší počet sloupců a místo umělých klíčů, které ve faktové tabulce tvoří odkazy na dimenze, obsahuje přirozené klíče těchto dimenzí (např. místo “IdZbozi” obsahuje přímo “KodZbozi”). Struktura tabulky vznikla díky sledování nejčastěji používaných dotazů v QDS nástroji, proto nad ní dotazy probíhají výrazně rychleji, než kdyby byla použita původní struktura DW, která vyžaduje použití mnoha `JOIN`ů.

Zbývající balíček “Nastav Flag Zpracování OLAPu”, nastaví do speciální tabulky “TSParametr” takový příznak, aby došlo ke zpracování OLAP vrstvy Intrastatu, a to až po úspěšném načtení do první vrstvy.

Ještě bych rád zmínil balíček “Zápis chyby do logu”, který je spuštěn v případě selhání zpracování jakéhokoli z balíčků v “IntrastatLoad”. Obsahuje následující kód, zajišťující, že v případě selhání ETL procesu bude možné chybu identifikovat, odstranit a dokončit load dat.

```
DECLARE @TypZaznamu char(1),
          @Priorita smallint,
          @Nazev varchar(250),
          @Popis varchar(8000),
          @EtlProcesId int
SELECT @TypZaznamu = 'C',
          @Priorita = 2,
          @Nazev = 'Identifikace místa, kde nastala chyba',
          @Popis = 'Podrobný nepovinný popis chyby',
          @EtlProcesId = 0,

exec PZapisChybu @TypZaznamu, @Priorita, @Nazev, @Popis, @EtlProcesId
```

5 INTEGRATION SERVICES[33]

Ačkoli se Data Transformation Services staly poměrně oblíbeným ETL nástrojem, který byl (a stále je) hojně využíván pro přenos dat nejen v rámci MS SQL Serveru 2000, v nové verzi databázového serveru je již nenajdeme. Místo nich SQL Server 2005 totiž obsahuje *SSIS* neboli *SQL Server Integration Services* (*Integrační služby*). Zde se ovšem nejedná, jak bývá zvykem, pouze o změnu názvu z marketingových důvodů. Integration Services jsou kompletně přepracované a dalo by se říci, že se jedná o zcela nový[34] ETL nástroj, navržený úplně od začátku. Vzhledem k tomu, že etapa ETL je při budování a provozu DW klíčová a DTS byl (je) jedním z nejjednodušších a nejintuitivněji ovladatelných nástrojů, vzbudil tento fakt mezi odbornou veřejností značný rozruch.

Důvody ke změnám jsou podle článku [32], který je k dispozici na webu Microsoftu, následující:

"*Ačkoli byl DTS velmi užitečný nástroj, měl některé limitace ve škálovatelnosti a snadném přesouvání balíčků mezi rozdílnými SQL Servery*".

Na úvod je třeba zdůraznit, že Integration Services nejsou pouze ETL nástrojem. Podle slov Donalda Farmera, programového manažera SSIS týmu, sice dokáže přesouvat miliony řádků mezi heterogenními datovými zdroji, ale jeho funkcionalita zde nekončí [27]. SSIS jsou totiž kompletní platformou pro integraci dat, obsahující grafické vývojové prostředí a nástroje pro správu, programovatelné objekty a aplikační programové rozhraní (API) [32].

5. 1 Architektura SSIS

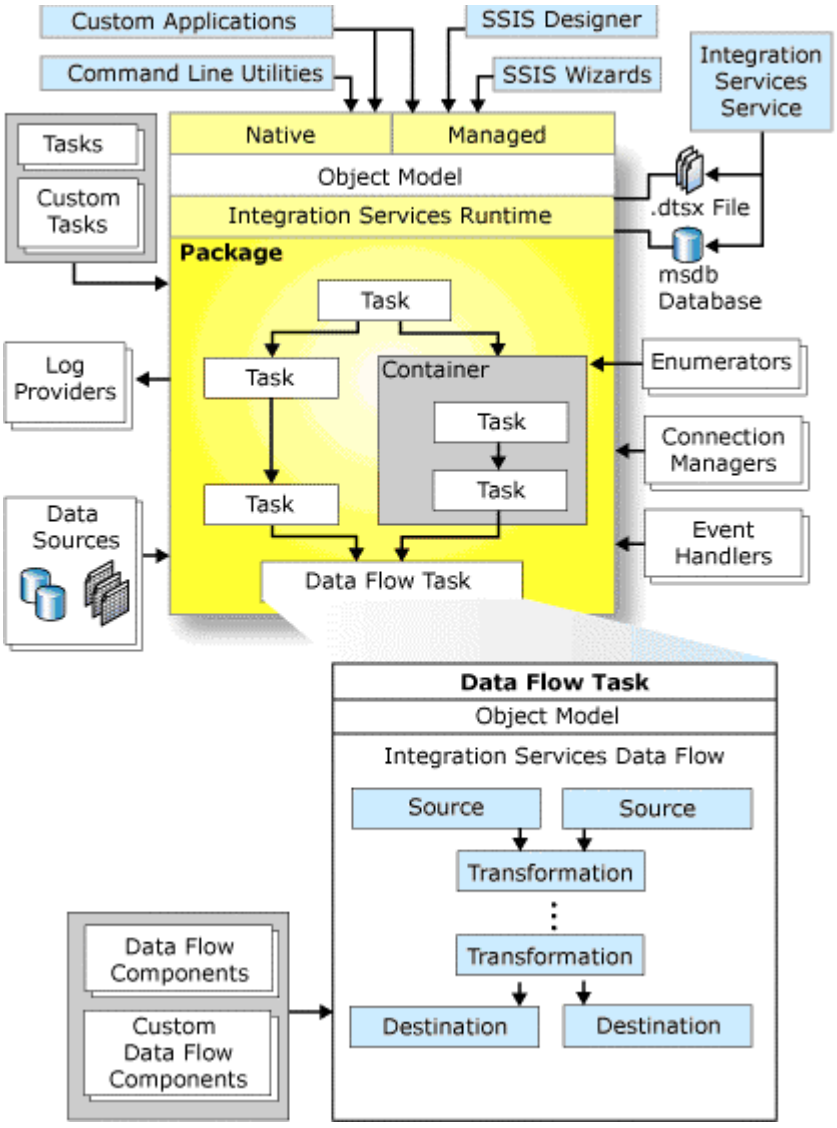
Následující obrázek názorně ukazuje architekturu Integration Services. Klíčovým prvkem SSIS je, stejně jako u DTS, balíček (*package*). Vytváří se pomocí nástroje *SSIS Designer*, průvodce *SSIS Wizard* nebo příkazové řádky. Je tedy možné využít jak vizuálního návrhu (řízený kód), tak balíčky programovat (nativní kód).

Na první pohled je zřejmý největší rozdíl mezi SSIS a DTS, totiž oddělení *procesních toků (control flow)* od *toků dat (data flow)*. O procesní toky, jako je uložení návrhu balíčku a správa ladění, logování, konfigurace, připojení a transakce, se stará tzv. *Data Transformation Runtime (DTR)*, zatímco *Data Transformation Pipeline (DTP)* nebo také *Data Flow Engine* řídí tok dat ze zdroje, přes transformace na místo určení.

Architektura pipeline (jádra SSIS) je založena na vyrovnávací paměti. To umožňuje extrémně rychlou manipulaci s daty. Veškerá data, ať už se jedná o strukturovaná, nestrukturovaná či XML, jsou překonvertována na tabulková (řádky a sloupce) a poté načtena do paměti. Díky tomu lze omezit nutnost použití Stage oblasti pro přenos menších objemů dat, kterou tradiční nástroje vyžadují téměř v každé části ETL procesu. Pro dosažení maximálního výkonu je (zejména u přenosu a transformace rozsáhlejších tabulek) samozřejmě více než vhodné Stage použít a Integration Services nabízejí dobrou podporu pro její implementaci [15].

[33] Kapitola byla napsána s využitím následujících zdrojů: [15], [21], [24], [27], [32].

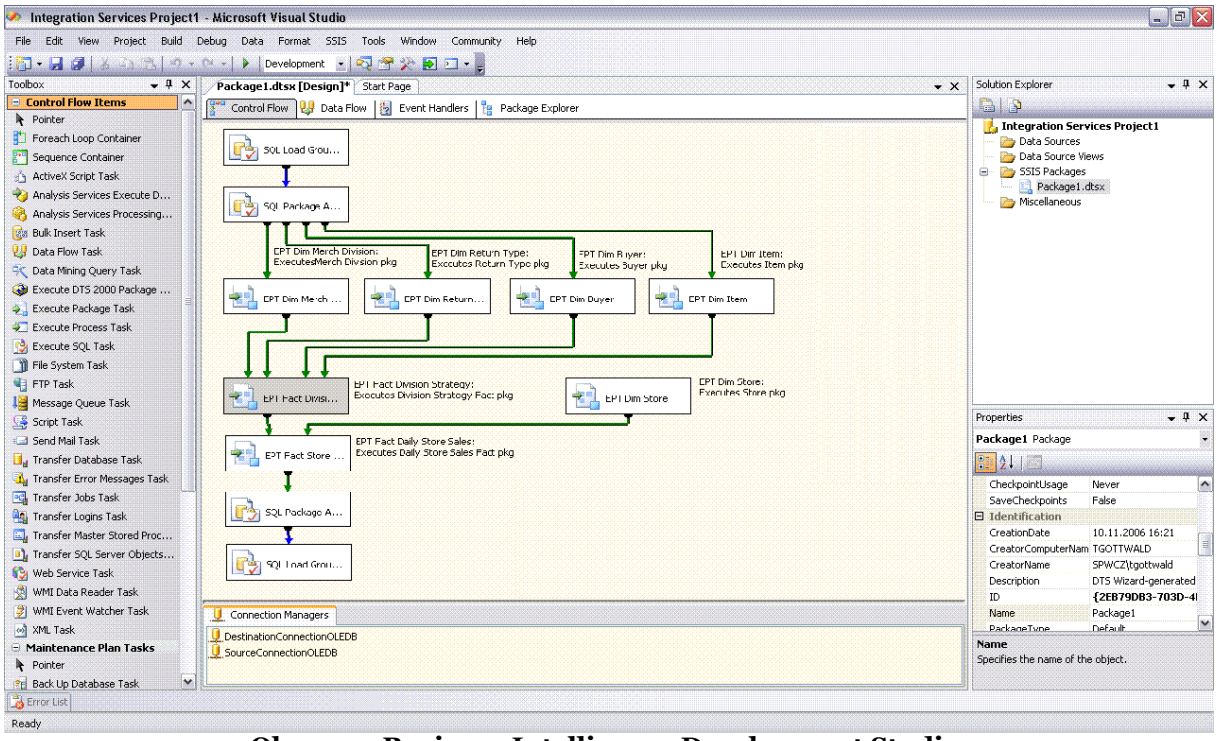
[34] Není divu, vždyť vývojový tým pod vedením Donald Farmera na něm pracoval téměř pět let.



Obr. 21 - Architektura Integration Services. Zdroj: [32]

5. 2 Business Intelligence Development Studio

Business Intelligence Development Studio (BIDS) je hlavním vývojovým nástrojem nejen pro Integrační služby, ale i pro *MS Reporting Services*, OLAP a Data mining – zkrátka pro celou oblast Business Intelligence v SQL Serveru 2005. Je založeno na *MS Visual Studiu .NET 2005* a jedná se o nástroj podporující "drag and drop" a vizuální modelování. Jeho pomocí lze budovat robustní a komplexní datové transformace, obsahují mnohonásobné připojení k heterogenním datovým zdrojům, komplexní posloupnosti úloh, datové transformace a událostmi řízenou logiku [32]. Velkou výhodou tohoto vývojového prostředí je, že se neváže k žádnému konkrétnímu SQL Serveru. Jinými slovy, balíček lze navrhnout v režimu offline a teprve poté jej spustit na serveru. To ve verzi 2000 nebylo možné, při vývoji balíčku v Enterprise Manageru bylo nutné připojení k instanci SQL Serveru [21].



Obr. 22 - Business Intelligence Development Studio.

Obrázek ukazuje, že hlavní okno vývojového prostředí je rozděleno na čtyři záložky [24]:

- *Control flow* (procesní tok),
- *Data flow* (datový tok),
- *Event handlers* (obsluha událostí),
- *Package explorer* (prohlížeč balíčků).

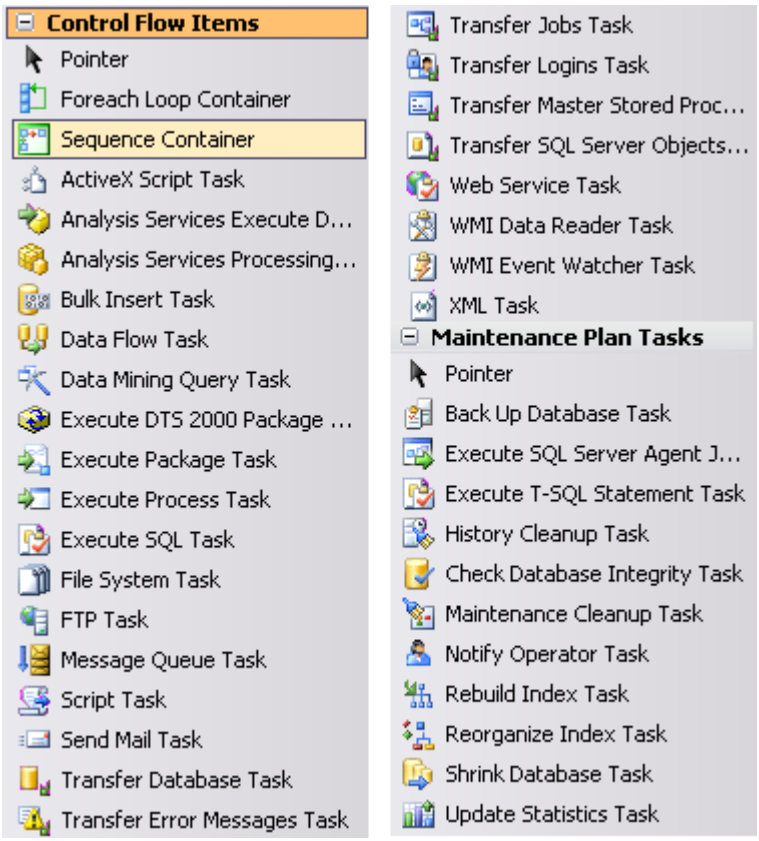
5. 2. 1 Control flow

Pomocí control flow lze definovat akce, které jsou vykonávány po spuštění balíčku. Podobně jako u DTS jsou k dispozici typy úloh, pomocí kterých lze provádět široké spektrum operací. Aby bylo možné modelovat tok procesů, je potřeba určit v jakém pořadí a za jakých podmínek se budou jednotlivé úlohy vykonávat. K tomu v SSIS (stejně jako v DTS) slouží *Precedence Constraints* neboli posloupnost operací. Zůstaly volby *Success* (po úspěšném dokončení), *Completion* (po dokončení) a *Failure* (po neúspěšném dokončení), ale nově je možné řídit workflow pomocí podmíněných výrazů. V drop-down boxu *Evaluation property* je možné nastavit:

- *Constraint* – tok procesů je určen volbou posloupnosti operací, tj. Success/Failure/Completion,
- *Expression* – pomocí výrazů a připravených funkcí lze definovat vlastní podmínky,
- *ExpressionAndConstraint* – před spuštěním musí být splněna podmínka výrazu i posloupnosti operací,
- *ExpressionOrConstraint* – specifikuje, jestli má být splněna podmínka výrazu nebo posloupnosti operací.

Pokud má úloha více Precedence Constraints (*Multiple Constraints*), je možné určit v jakém mají být vztahu. Mohou být ve vztahu *logický AND* (výsledek všech musí být *True*) nebo *logický OR* (alespoň jeden výsledek musí být *True*).

Nabídku úloh najdeme v toolboxu na levé straně vývojového prostředí. Úlohy je možné rozdělit do několika logických skupin. Některé z nich jsou stejné jako v DTS 2000, jiné byly přepracovány a vylepšeny a přibyl také velký počet naprosto nových úloh, které dále rozšiřují možnosti nástroje.



Obr. 23 - Control Flow Toolbox

Mezi zbrusu nové prvky patří hned první tři položky ze seznamu, tzv. *containers* (kontejnery), které umožňují jednoduché vytváření cyklů a/nebo seskupení několika úloh do logických skupin, podobně jako Case nástroje typu *Sybase Power Designer* nebo *Enterprise Architect*. Obsah kontejneru je možno kliknutím skrýt či zobrazit. Celý návrh ETL procesů se tak stává daleko přehlednějším. Nejčastější využití najdou u Data Flow úloh, kde jsou data zpracovávána cyklicky. Do skupiny kontejnerů patří [24]:

- *For Loop Containter* – dovoluje vytváření opakovaných cyklů, řízených počítadlem cyklu. Je definována počáteční a koncová podmínka a inkrementace počítadla. Jedná se o ekvivalent *for cyklu* známého z programovacích jazyků.
- *Foreach Loop Container* – konec cyklu je zde dán enumerátorem, například vyčerpáním prvků dané množiny. Najde využití kupříkladu pokud je potřeba zpracovat všechny soubory v adresáři.
- *Sequence Container* – slouží k zapouzdření několika bloků diagramu Control Flow.

Další skupina úloh pracuje se soubory, adresáři, umožňuje stahovat webový obsah či načítat obsah XML dokumentů. Nepracují s vlastním obsahem dat, ale mají za úkol data připravit. SSIS poskytuje pro přípravu dat následující úlohy:

- *File System Task* – je novinkou v SQL Serveru 2005 a umožňuje operace typu kopírování, přesouvání, přejmenovávání a mazání souborů, případně adresářů, které lze navíc i vytvářet. Další funkcí je např. nastavování atributů operačního systému. Dříve bylo nutné využívat pro tyto operace ActiveX Script Task, což s sebou přinášelo tvorbu velkého množství skriptů.

- *FTP Task* – zatímco předchozí verze umožňovala pouze obdržet či poslat soubory přes FTP, SSIS poskytuje podporu i pro mazání souborů či mazání a vytváření adresářů a to jak lokálních, tak vzdálených.
- *Web Service Task* – je zbrusu nová úloha, která se dokáže připojit na webovou službu a obdržená data uložit do souboru či proměnné. Využití najde zejména v případech, kdy je potřeba použít data uložená na webu třetích stran.
- *XML Task* – poskytuje široké možnosti manipulace s XML soubory za běhu a představuje tak další z mnoha úloh, které v DTS implementovány nebyly. Prostřednictvím *XML Task Editoru* vývojář zvolí, zda chce XML dokument validovat proti *Document Type Definition (DTD)* či *XML schématu (XSD)*, provádět *XSLT transformace*, vykonávat *XPATH* dotazy, slučovat dva XML dokumenty do jednoho, porovnávat dva dokumenty nebo zda vytvoří nový XML soubor z výsledků porovnání.

Jiná část úloh se orientuje především na komunikaci s procesy, operačním systémem a službami, kde jsou spouštěny balíčky, zajišťuje posílání e-mailů, atd. Jedná se o tyto workflow úlohy:

- *Execute Package Task* – byl oproti DTS vylepšen o volbu *ExecuteOutOfProcess*, která zajistí spuštění balíčku jako samostatného procesu s vlastní přidělenou pamětí, což je z hlediska využití paměti serveru náročnější, ale na druhou stranu lze dosáhnout vyššího výkonu. Další změny byly provedeny také u balíčků, které jsou ve vztahu parent-child. Nyní již rodičovský balíček „nevnucuje“ své proměnné dětskému, naopak dětský přistoupí do rodičovského a nastaví konfigurační hodnoty.
- *Execute DTS 2000 Package Task* - je jednou z cest, jak upgradovat ETL vrstvu realizovanou v DTS na SSIS, umožňuje totiž spouštět balíčky vyvinuté v DTS. Další možností je použití Migration Wizard, kterému se budu věnovat v další části práce. Nicméně již teď bych rád předeslal, že i když jsou oba zmíněné postupy poměrně pohodlné, nejlepší (i když nejpracnější) možností je ta třetí, to znamená ručně předělat balíčky v Integration Services. Jen tak budou využity obrovské možnosti SSIS.
- *Execute Process Task* – slouží ke spouštění externích aplikací uvnitř balíčku. Oproti DTS byl podstatně vylepšen o robustní obsluhu chyb a veškeré chybové hlášení i jiné výstupy z příkazového řádku je možné uchovat pro pozdější zpracování.
- *Message Queue Task* – odesílá a přijímá zprávy MSMQ, což lze využít např. pro vzájemnou komunikaci balíčků za běhu. Mohou si posílat text, soubory či proměnné.
- *Send Mail Task* – v SSIS již neposílá maily přes protokol MAPI, což znamenalo nutnost mít na serveru nainstalován MS Outlook, ale zvládne standardní SMTP protokol. Typicky bývá tato úloha zařazována mezi operace, které jsou vykonávány při selhání ETL procesu, může např. poslat e-mail administrátorovi.
- *WMI Data Reader Task* – dokáže spustit dotazy typu WQL pod Windows Management Instrumentation. Takto lze zjistit např. dostupný HW, seznam nainstalovaných aplikací nebo číst data z event-logů.
- *WMI Event Watcher Task* – poskytuje SSIS možnost čekat na určitou WMI událost, která má nastat v operačním systému. Může se jednat o start nějaké služby, volnou kapacitu procesoru, volnou paměť nebo lze sledovat adresář a čekat, zda se v něm neobjeví požadovaný soubor.

Pomocí následující sady úloh lze přistupovat k datům a objektům pod správou SQL Serveru :

- *Bulk Insert Task* – načítá data pomocí příkazu `BULK INSERT`.
- *Execute SQL Task* – stejně jako v DTS umožňuje spustit jeden nebo více SQL příkazů nebo uloženou proceduru. Nově ale umožňuje i spuštění SQL kódu, který je uložen v souboru. Je zde možnost nastavení *time-outu*[35], a také volba, v jakém formátu má být výstup. Lze nastavit *single row* (jeden řádek), *full result set* nebo *XML* formát.

[35] Time-out je doba (v sekundách) po kterou se databázový stroj snaží o vykonání dotazu.

Příjemné je také, že výstup se dá použít i jinde v balíčku. Tímto způsobem lze například zkontrolovat, zda byl vrácen správný výsledek.

- *Transfer Database Task* – přenese databázi z jedné instance SQL Serveru na druhou, jak ostatně název napovídá. Přeneseny jsou i loginy, role a práva k objektům, což je mnohdy zbytečné zatěžování systémů, takže bývá výhodnější přenést jednotlivé objekty.
- *Transfer SQL Server Objects Task* – dokáže mezi instancemi SQL Serveru pomocí *CopyAllObjects* přenést tabulky, pohledy, uložené procedury, atd. Indexy, triggery, primární a cizí klíče je k přenosu třeba vybrat individuálně.
- *Transfer Error Messages Task , Transfer Job Task, Transfer Logins Task, Transfer Master Stored Procedures Task* – zajišťují přenos uživatelsky definovaných chybových zpráv, úloh pro SQL Server Agenta, přístupových práv a uživatelsky definovaných procedur mezi dvěmi instancemi SQL Serveru.

Integration Services obsahují také dvě úlohy rozšiřující možnosti balíčků o skriptový kód:

- *ActiveX Task* - má stejné možnosti jako v DTS, např. podporu více skriptovacích jazyků a možnost jejich dodatečné instalace.
- *Script Task* - umožní přístup do prostředí MS Visual Studio for Applications (VSA) pro vývoj a spouštění skriptů v jazyce VB.NET. *Script Task* poskytuje širší možnosti než *ActiveX Task*, např. podporu pro přidání *breakpoints*[36] do kódu, snadné předání proměnných do skriptu a jeho předkompilování pro rychlejší vykonání operací, které zajišťuje.

SSIS také disponují nástroji pro přístup do MS Analysis Services:

- *Analysis Services Process Task* – umožňuje zpracování OLAP kostky, dimenze nebo Data miningového modelu.
- *Analysis Services DDL Task* – spustí úlohu typu DDL[37] v prostředí analytických služeb. Její pomocí je tedy možné např. vytvořit, odstranit nebo upravit kostku. Je to vlastně ekvivalent Execute SQL Tasku pro Analysis Services, není však podobně robustní. Výsledky například nelze předat do proměnné.
- *Data Mining Query Task* – prostřednictvím Analysis Services povoluje spouštění prediktivních dotazů na Data miningových modelech v Analysis Services a použití výsledků jako datového zdroje.

Kromě samotných úloh určených ke správě procesů najdeme v nabídce i tzv. *Maintenance Plan Tasks*, což jsou entity pro úlohy sloužící ke správě a údržbě:

- *Back Up Database Task* – umožňuje zálohovat databáze.
- *Check Database Integrity Task* – kontrolují strukturální integritu databáze a integritu objektů v databázi.
- Execute SQL Server Agent Job Task – spustí úlohu SQL Server Agenta.
- *Execute T-SQL Statement Task* – slouží ke spuštění bloku kódu jazyka Transact SQL.
- *History Cleanup Task* – vymaže data týkající se monitorování historie.
- *Rebuild Index Task* – přebuduje indexy ve všech databázích, ve vybraných databázích, tabulkách nebo pohledech.
- *Reorganize Index Task* – optimalizuje indexy.
- *Shrink Database Task* – pokud velikost databáze vzroste nad určenou hranici, zhustí její obsah.
- *Update Statistics Task* – aktualizuje statistická data.

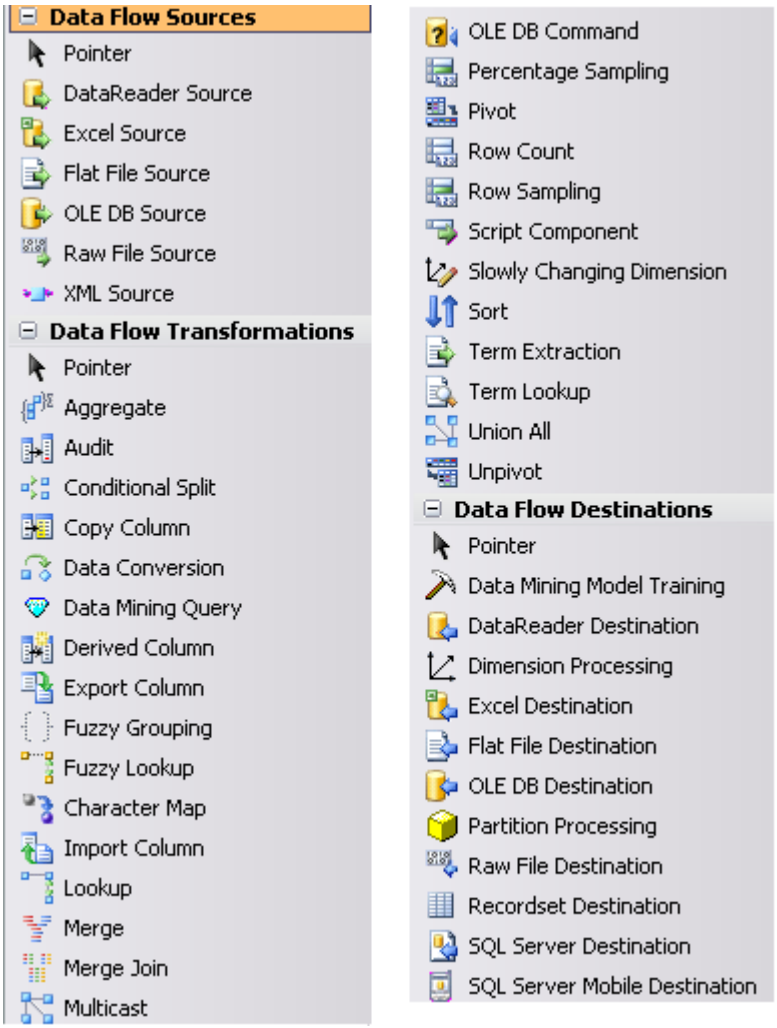
[36] Pojem bude vysvětlen v samostatné podkapitole.

[37] DDL (Data Definition Language) neboli jazyk pro definici dat slouží v relačních databázích k vytváření relačních objektů jako jsou např. tabulky nebo indexy.

5. 2. 2 Data Flow

Jako poslední uvedu ve výčtu úloh z nabídky Control Flow tu nejdůležitější – blok *Data Flow*. Je orientován na práci s daty a v závislosti na režimu řádky s daty produkuje nebo konzumuje. Jedná se o úlohu natolik komplexní, že pro její diagram byla vyčleněna zvláštní záložka pracovní plochy Business Intelligence Development Studia, odkud je také dostupný toolbox s nabídkou Data Flow úloh. Ty je možné rozdělit do třech základních kategorií:

- *Data Flow Sources* – datové zdroje,
- *Data Flow Transformations* – transformace dat,
- *Data Flow Destinations* – cílové uložiště dat.



Obr. 24 - Data Flow Toolbox

Pro každý ETL proces je nutné určit zdroj dat. Definované zdroje se zobrazí nejen v Data Flow diagramu, ale i v tzv. *Connection Manageru*, který se nachází pod hlavním oknem vývojového prostředí BIDS. Connection Manager umožňuje znovupoužití zdrojů a cílových uložišť dat. I případné změny je tedy nutné provést pouze na jednom místě. SSIS nabízí šest alternativ datových zdrojů:

- *OLE DB Source* – je zřejmě nejčastěji využívaným zdrojem, umožňující připojit jakýkoli zdroj podporující OLE DB standard.
- *Excel Source* – připojí tabulkový procesor MS Excel.
- *Flat File Source* – textový soubor, ve kterém jsou data v řádcích oddělena čárkami nebo tabulátorem, případně jsou k oddělení sloupců určeny pevné počáteční a

koncové body. Další možností je volba *Ragged Right*, kde mají sloupce, s výjimkou posledního, který je oddělen čárkami či tabulátorem, pevnou šířku. Data ze zdroje tohoto typu jsou načítána velmi rychle, na druhou stranu vyžadují náročnější mapování sloupců.
- *Raw File Source* – je speciální typ textového souboru, který byl vytvořen pomocí úlohy *Raw File Destination*. Přináší podstatně rychlejší načítání dat, ale za cenu menší flexibility.
- *XML Source* – zajišťuje připojení lokálních či vzdálených XML souborů.
- *Data Reader Source* – umožňuje vytvoření .NET poskytovatele a jeho použití uvnitř balíčků.

Úlohy určené k transformaci dat jsou pro vývojáře ETL možná nejpříjemnější z celé řady inovací, které Integration Services přinesly, protože jim ušetří spoustu času. Pro typické ETL operace, jako jsou například lookup, konverze datových typů, odhalování duplicit a podobně, totiž nemusí psát složité skripty, ale stačí jim vybrat si některou z nabízených úloh. Transformace jsou navíc, jak jsem již v úvodu kapitoly uvedl, vykonávány v paměti a není tudíž třeba vytvářet Stageovací tabulky v každém kroku procesu. Samozřejmě při velkých objemech dat je použití Stage nutné, stejně tak jako je pro komplikované a unikátní operace s daty potřeba psát transformační skripty.

- *Aggregate* – umožňuje na množinu dat, vstupujících do transformace, aplikovat agregační funkce jazyka T-SQL. Podporovány jsou tyto operace:
  - *Group by* – rozdělení záznamů do skupin,
  - *Average* – průměr ze sloupce numerických dat,
  - *Count* – počet záznamů v množině dat,
  - *Count Distinct* – počet unikátních záznamů,
  - *Minimum* – nejnižší numerická hodnota,
  - *Maximum* - nejvyšší numerická hodnota,
  - *Sum* – součet numerických hodnot.
- *Audit* – poskytuje operativní získávání dat z prostředí, kde je *Data Flow* úloha spuštěná. Zejména v současné době, kdy je kvůli regulačním předpisům, jako jsou např. *HIPPA* a *Sarbanes-Oxley (SOX),* nutná schopnost doložit, kdo záznam vložil a kdy, bude auditovací úloha vítaným pomocníkem. Díky ní lze snadno zjistit informace o balíčku (identifikátor, název, verze, čas spuštění), serveru či počítači, uživateli a úloze asociované s auditem (název a identifikátor).
- *Conditional Split* – přináší velmi snadný způsob rozdělení množiny dat podle zadaných kritérií. Pro definování podmínek rozdělení je k dispozici široká škála matematických, textových, datetime a jiných funkcí a řada matematických i logických operátorů.
- *Copy Column* – přidává nové sloupce, které jsou kopiemi stávajících. Kopií, vytvořených před transformací, lze s výhodou využít pro porovnání původních a transformovaných dat.
- *Data Conversion* – provádí převody mezi datovými typy stejně jako příkazy `CAST` a `CONVERT` jazyka T-SQL.
- *Data Mining Query* – spustí dotaz nad data miningovým modelem a výsledek přidá do *Data Flow*. Tímto způsobem je např. možné místo neznámých hodnot doplnit hodnoty předpokládané či přidat do tabulky sloupec, který bude vyjadřovat pravděpodobnost nastání určitého jevu.
- *Derived Column* – slouží k vytvoření nového sloupce, jehož obsah je odvozen ze stávajících sloupců za pomoci nabízených operátorů a funkcí. Taková transformace se hodí například v případech, kdy známe počet kusů a cenu výrobků v objednávce a potřebujeme přidat sloupec s celkovou cenou nebo máme-li sloupce jméno a příjmení a potřebujeme je spojit do jednoho.
- *Export Column* – dokáže exportovat obrázek či soubor z Data Flow. Na rozdíl od jiných transformací při vytváření souboru není třeba definovat cílové uložiště.

- *Fuzzy Grouping* – slouží k nalezení vzorů, které mohou reprezentovat duplikovaná data. Pojem *fuzzy*[38] znamená „rozmazaný“ nebo „neostrý“ a *neostré duplicity* jsou záznamy, které nejsou totožné (*exaktní duplicity*), ale mají určité společné znaky, reprezentují jeden objekt v realitě a je tudíž nutné sloučit je do jednoho řádku. Příkladem duplicity tohoto typu mohou být záznamy „J. Vrchlického 2733/35, 434 00 MOST“ a „Vrchlického ul. 35, 434 00 MOST“.
- *Fuzzy Lookup* – je operace podobná SQL příkazu `JOIN` (spojení) fungující ovšem na základě fuzzy logiky. Nespojuje tedy shodné záznamy, ale ty, které by mohly být považovány za shodné. Lze také nastavit hranici, podle které SSIS určí, zda se jedná o záznam podobný nebo shodný, a to pomocí parametrů *Similarity* a *Confidence*. *Similarity* (podobnost) je číslo od 0 do 1, kde 1 znamená naprosto shodný záznam. *Confidence* (důvěra), kterou lze určit na stejném intervalu, neporovnává pouze jeden výraz oproti jinému, ale vybrané spojení oproti všem ostatním možným spojením. Díky úloze *Fuzzy Lookup* lze několikanásobně zvýšit počet spárovaných záznamů, ale je vhodné ho použít až po klasickém lookupu, protože používá speciální indexování, což má samozřejmě negativní vliv na výkon.
- *Charakter Map* – vykonává překlady znaků typu: z velkých na malá písmena a opačně, změnu národních jazykových pravidel a, což zejména čeští vývojáři jistě ocení, změnu japonských znaků z Hiragana do Katakana stylu a převody mezi tradiční a zjednodušenou čínštinou. Překlad lze vykonat na stávajícím sloupci nebo přidat sloupec nový.
- *Import Column* – je opak úlohy Export Column – tzn. importuje obrázek nebo soubor z adresáře do Data Flow.
- *Lookup* – nebyl v DTS veden jako samostatná úloha, ale byla mu věnována jedna ze záložek Data Transform Tasku. Stejně jako ve verzi 2000 by měl být využíván střídmě, protože trvá déle než klasický `JOIN`.
- *Merge* – umožňuje spojit data pocházející z různých větví diagramu do jednoho výstupu. Data ale musí být setříděna pomocí příkazu `ORDER BY` nebo úlohy *Sort* a metadata ve větvích musí být shodná (např. sloupec „KodZakaznika“ nemůže mít v jedné větvi datový typ `integer` a v druhé `varchar`.
- *Merge Join* – přináší, v souladu s jedním z ústředních témat SSIS, kterým je minimalizace nutnosti psaní kódu, použití úlohy místo T-SQL příkazů pro spojení `INNER JOIN` a `OUTER JOIN`. *Merge Join* je vhodné použít pro spojení dat z různorodých zdrojů, nikoli pokud je třeba spojit tabulky z jedné databáze – zde je rychlejší a efektivnější použít zmiňované T-SQL příkazy.
- *Multicast* – umožňuje odesílání dat z jednoho zdroje do více uložišť.
- *OLE DB Command* – spustí SQL příkaz na každý řádek v Data flow. Tato úloha může při velkých objemech dat značně prodloužit celý ETL proces, proto je její využití nutné důkladně zvážit.
- *Percentage* a *Row Sampling* – umožňují náhodný výběr podmnožiny dat, kterou je možné využít pro testování nebo trénování data miningových modelů. Velikost podmnožiny je možné určit procentem z celku, resp. počtem řádků.
- *Pivot* a *Unpivot* – úloha *Pivot* denormalizuje data, aby mohla být zobrazena ve formě tzv. *kontingenční tabulky*. *Unpivot* provádí opačný proces, tj. normalizaci.
- *Row Count* – předá do proměnné počet řádků vstupujících do transformace. Využití najde např. pro zjišťování počtu záznamů načtených nebo naopak nenačtených do skladu.
- *Script Component* – poskytuje možnost vytvořit pomocí skriptů vlastní transformace, zdroje nebo uložiště dat.

[38] Průkopníkem fuzzy logiky byl Lotfi Zadeh z Kalifornské univerzity v Berkeley, který rozmazal ostré kontury klasické logiky, založené na Aristotelovu zákonu vyloučení třetího (který říká, že tvrzení může být buď pravdivé nebo nepravdivé), svým převratným článkem Fuzzy množiny, kde píše: *„…lidé mají pozoruhodnou schopnost rozumně se rozhodovat v situacích charakterizovaných nejistotou a nepřesností. Dokážeme rozumět zkomolené řeči, rozluštit lajdácké písmo… Přitom neprovádíme žádné složité výpočty v běžném smyslu slova. Zpracováváme informaci, což právě dělají počítače, ale objekty našich úvah obecně nejsou čísla, nýbrž rozmazané fuzzy obrazce bez ostře vymezených hranic*“ [5].

- *Slowly Changing Dimension* – je jednou z nejzajímavějších úloh v Integration Services, usnadňuje totiž aktualizaci stávajících a vkládání nových záznamů do dimenzí. Po umístění úlohy pro SCD do *Data Flow* je spuštěn průvodce *Slowly Changing Dimension Wizard*. V prvním kroku je třeba definovat umístění dimenzionální tabulky, mapování vstupních polí ze zdroje do dimenzionální tabulky kvůli porovnání a určit tzv. *business key*. Jedná se o přirozený klíč nebo klíčové hodnoty pro obchod, tedy pole nepodléhající změnám. V dalším kroku průvodce je třeba určit, jaké sloupce a jakým způsobem se budou měnit:
  - *Fixed Attribute* – hodnoty nebudou podléhat změnám. Pokud záznam ze zdroje nebude v dimenzi nalezen, lze zvolit, zda má být ignorován nebo má transformace skončit chybou,
  - *Changing Attribute* – záznam bude aktualizován podle zdroje,
  - *Historical Attribute* – přidá záznam včetně jeho platnosti, jenž je určena buď počátečním a koncovým datem nebo pomocí voleb *True/False*, resp. *Current/Expired*.

  Následuje závěrečná volba *Inferred Member selections*, která umožňuje naplnění dimenzionální tabulky v případech, kdy existuje vazba na faktovou tabulku, ale některé atributy dimenze nejsou dočasně k dispozici. Po tuto dobu jsou vedeny jako `NULL` hodnoty, případně nesou booleovský příznak odlišující neznámé atributy.
- *Sort* – umožňuje třídit data podle vybraného sloupce. Vzhledem k tomu, že velké množství jiných úloh vyžaduje na vstupu již setříděné hodnoty, bude *Sort* využívána poměrně často.
- *Term Extraction* – je zajímavá úloha, která ukazuje výskyt klíčových slov v datech. Po definování zdroje je vrácen výsledek v podobě dvou sloupců - seznamu výrazů a počtu jejich výskytů nebo tzv. *TDIDF skóre*[39]. Všechny zdrojové sloupce jsou zahozeny, proto je potřeba zařadit před *Term Extraction* úlohu *Multicast* a zálohovat. Úloha bohužel pracuje pouze s anglickými slovy a jazykovými pravidly.
- *Term Lookup* – používá stejné algoritmy a statistické modely jako *Term Extraction*, ale funguje odlišně. K hledanému seznamu výrazů připojí záznamy, ve kterých se požadované výrazy vyskytují.
- *Union All* – sdruží data z více zdrojů. Nevyžaduje třídění ani shodu metadat jako *Merge Join*, navíc při integraci dat z více než dvou zdrojů je jeho využití výhodnější.

Abych výčet položek, nabízených toolboxem záložky *Data Flow*, zkompletoval, uvedu ještě *Destinations* neboli cílová uložiště dat:

- *Data Mining Model Training* – umožňuje učení modelu pomocí dat z *Data Flow*. Data ovšem musí být setříděná, proto je vhodné použít nejprve úlohu *Sort*.
- *DataReader Destination* – poskytuje data aplikacím, které mohou využívat DataReader rozhraní, např. MS Reporting Services.
- *Dimension* a *Partition Processing* – umožňuje naplnit a procesovat dimenzi (resp. partition) v Analysis Services. Na výběr je aktualizace, plné nebo inkrementální plnění.
- *Excel Destination*.
- *Flat File Destination* – lze zvolit oddělovače nebo pevnou šířku sloupce a přidat vlastní záhlaví.
- *OLE DB Destination*.
- *Raw File Destination*.
- *Recordset Destination* – slouží k použití ADO record setu mimo transformaci, nepodporuje ovšem generování chyb do logu, jako ostatní úlohy.
- *SQL Server Mobile Destinations* – uložiště optimalizované pro SQL Server, využívající rychlé vkládání pomocí `BULK INSERT`. Lze ho ale využít pouze pokud je

[39] *TDIDF (Term Frequency and Inverse Document Frequency)* je statistický ukazatel, který se vypočítává následujícím vzorcem:
`TDIDF = (počet výskytů výrazu) * log(počet řádků celkem) / (počet řádků s výskytem výrazu)`

balíček spuštěn na stejném serveru, kde běží SQL Server. Výstup je vhodný pro nasměrování dat do Pocket PC zařízení.

5. 2. 3 Event Handlers

Záložka *Event Handlers* neboli obsluha událostí poskytuje ETL vývojářům obrovské možnosti. Mohou totiž zahrnout do balíčků reakce na různé události, které mohou nastat během vykonávání. Jedná se o mocný nástroj především pro obsluhu chyb. V SQL Serveru 2000 byla k dispozici pouze volba *On Failure*, kterou ovšem bylo nutné vyřešit pro každý balíček zvlášť. Integration Services místo ní přinášejí událost *OnError*, kterou je možné použít globálně skrze celý balíček a shromáždit tak chybová hlášení do jednoho uložiště, popřípadě je poslat e-mailem administrátorovi. K dispozici je celá řada dalších událostí:

- *OnError* - pokud nastane chyba,
- *OnExecStatusChanged* – když se status vykonávané úlohy nebo balíčku změní,
- *OnInformation* - pokud se objeví informace v *Progress tabu*,
- *OnPostExecute* – v případě, že je dokončeno vykonávání úlohy či kontejneru (vhodné použít např. pro vymazání nepotřebných pracovních tabulek),
- *OnPostValidate* – po dokončení validace úlohy,
- *OnPreExecute* – před vykonáním úlohy (výhodné využít pro kontrolu vstupních hodnot),
- *OnPreValidate* – před validací úlohy,
- *OnProgress* – jakmile se začne úloha vykonávat,
- *OnQueryCancel* – po stornování dotazu,
- *OnTaskFailed* – pokud úloha selže,
- *OnVariableValueChanged* – v případě, že se za běhu změní hodnota proměnné,
- *OnWarning* – objeví-li se varování.

5. 2. 4 Package Explorer

Poslední ze záložek dostupných z hlavního okna vývojového prostředí je záložka *Package Explorer*, která, jak název napovídá, slouží k prohlížení obsahu balíčků. Na jednom místě jsou zobrazeny všechny úlohy, kontejnery, spojení, obsluhy událostí, proměnné a transformace zahrnuté v balíčku. Jednotlivé objekty je možné nejen prohlížet, ale i editovat.

5. 2. 5 Checkpoints

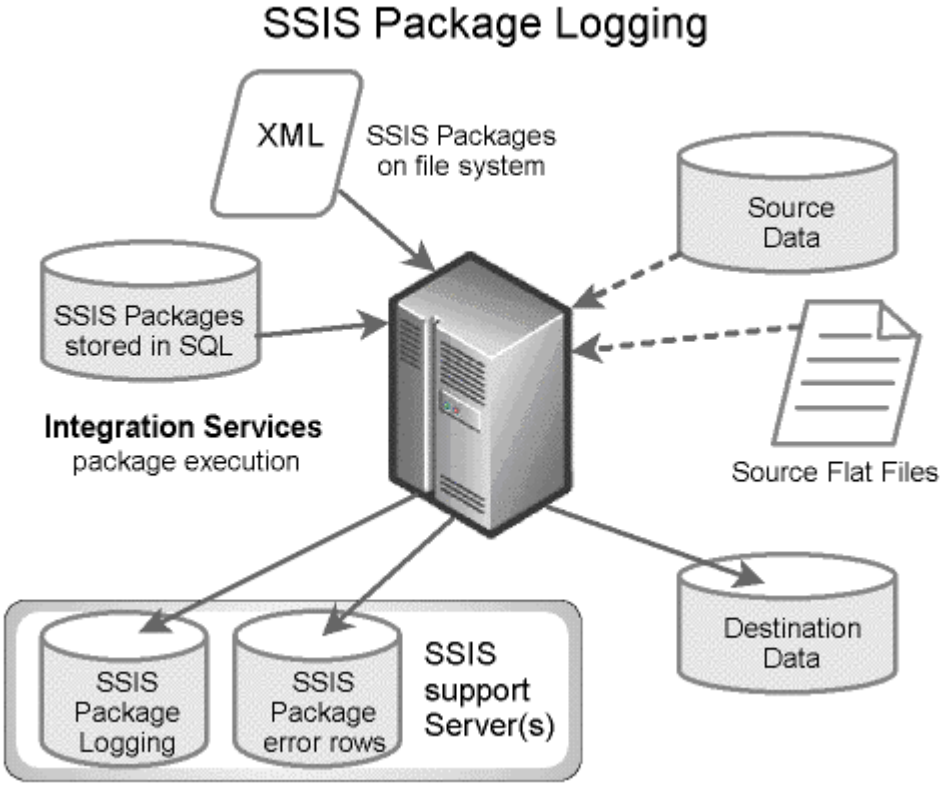
Kvůli nedostatečné schopnosti DTS reagovat na chybové události, byla do SSIS přidána záložka Event Handlers. Jenže dalším podstatným neduhem DTS při selhání balíčku bylo, že se celý balíček musel vykonávat znovu, úplně od začátku. A to při obrovských objemech dat, které jsou v dnešních podnicích obvyklé, způsobí značnou ztrátu drahocenného času. Proto byly do Integration Services implementovány tzv. *Checkpoints* neboli kontrolní body. Pokud je balíček nastaven na používání těchto kontrolních bodů (volba *SaveCheckpoints*), informace o jeho vykonávání jsou zapisovány do speciálního souboru a jestliže takový balíček selže, není důvod ho vykonávat celý znovu. Pokračuje se Control Flow úlohou, která selhala. To znamená, že se kontrolní body vztahují ke Control Flow, nikoli k Data Flow.

5. 2. 6 Logování

Jak ukázal příklad z praxe[40], v DTS museli ETL vývojáři používat vlastní skripty, pokud chtěli mít informace o vykonávání jednotlivých úloh. Microsoft pro ně v SSIS připravil užitečnou pomůcku, a sice logování (volba *Log Events* v menu *SSIS*). Informace o jednotlivých událostech jsou totiž zapisovány do logu, který může být ve formě:

[40] Zápis do tabulky „TSEtlLog“.

- *SQL Profiler Log Provider* – kombinuje logovací data s informacemi o výkonu systému a poskytuje tak podklady pro analýzu vlivu vykonávání balíčku na chod systému a pro odstraňování závad, jako jsou např. neočekávaně dlouhé časy vykonávání balíčků či úloh.
- *SQL Server Log Provider* – zapisuje logovací data do tabulky `sysdtslog90` v databázi a umožňuje centrálně uložit logovací data od všech SSIS balíčků, vykonávaných v rámci jednoho nebo více systémů. Informace z tabulky lze získat SQL dotazem.
- *Windows Event Log Provider* – je vhodné zvolit, pokud je k monitorování provozu SQL Serveru použit software typu *Microsoft Operations Manager (MOM)*. Jsou sem zapisovány základní informace i v případě, že logování není nastaveno.
- *XML File Log Provider* – umožňuje uložení logu ve formě XML souboru, který lze, po nezbytných XSLT transformacích, zobrazit jako webovou stránku. XML formát je také nejlepší pro sdílení dat a pro konsolidaci logovacích informací z více zdrojů.
- *Text File Log Provider* – vede log jako textový soubor.
- *Cystom Log Provider* – poskytuje organizacím možnost vytvoření takového log poskytovatele, který bude vyhovovat přesně jejich potřebám.



Obr. 25 - Logování v SSIS. Zdroj [32]

Logování je možné nastavit pouze pro některé nebo všechny úlohy a kontejnery, i pro některé nebo všechny události. Úlohy a kontejnery dědí nastavení logu po svých rodičovských kontejnerech.

5. 2. 7 Breakpoints

Dalším z nových pomocníků, kteří usnadní vývojářům práci, jsou tzv. breakpoints, což by se dalo přeložit jako bod zlomu nebo místo přerušení. Breakpoint lze umístit kamkoli do Control Flow nebo do kódu Script Task a ladit projekt, podobným způsobem, jako to umožňují jiné vývojářské nástroje. Užitečné je např. sledování hodnoty proměnných během vykonávání balíčků.

5. 2. 8 Škálovatelnost

Škálovatelnost je schopnost aplikace efektivně využívat více zdrojů, aby mohla podat vyšší výkon [4]. To ale neznamená pouze přidělení více zdrojů, naopak někdy může být výhodnější mít přiděleno méně. V úvodu této kapitoly jsem citoval oficiální vyjádření společnosti Microsoft, která právě nedostatky ve škálovatelnosti uvedla jako jeden z důvodů pro kompletní přepracování DTS. V Integration Services je možné škálovat např. přidělenou paměť. Data jsou totiž téměř výhradně zpracovávána v paměti, což sice eliminuje čas, který by byl jinak stráven čtením a zapisováním dat, ale na druhou stranu pro velké objemy a náročné transformace dat je nutné použít velkou část virtuální paměti. Ta je v 32-bitových operačních systémech Windows defaultně nastavena na 2 GB[41], proto je potřeba konfigurovat balíčky tak, aby ji využívaly co nejefektivněji, např. nastavením vlastnosti *ExecuteOutOfProcess* na *True*.

5. 3 Průvodci

Kromě Business Intelligence Development Studia, které je primárním vývojovým prostředím pro tvorbu balíčků, obsahují Integration Services řadu průvodců:

- *Import and Export Wizard* – slouží, stejně jako v DTS, k jednoduchým přenosům dat, při kterých se neprovádějí složitější transformace. Nabízí ale stejně širokou paletu zdrojů, jako BIDS. Navíc je zde možnost balíček uložit, znovu ho v Development Studiu otevřít a editovat.
- *Configuration Wizard* – průvodce nastavením balíčků, poskytující flexibilní metodu dynamické konfigurace balíčku za běhu. Takto nastavený balíček je možné spouštět v různých prostředích, protože operační parametry typu umístění souboru nejsou v balíčku „napevno“, ale jsou načteny až v průběhu vykonávání.
- *Package Installer Wizard* – je určen k instalaci balíčků do souborového systému nebo do databáze SQL Serveru. Po odladění balíčku na vývojářském počítači je možné spustit Installer Wizard v cílovém systému a pomocí průvodce tam balíček nainstalovat a nastavit.
- *Migration Wizard* – je po úloze Execute DTS 2000 Package Task a ručním předělání třetím způsobem migrace DTS balíčků na SQL Server 2005. Použití průvodce je z těchto možností nejsnazší, ale na druhou stranu automaticky zvládne konvertovat pouze DTS balíčky využívající jen standardní úlohy a transformace.

Kromě průvodců a BIDS lze manipulovat s balíčky pomocí SQL *Server Management Studia*[42], které nahradilo *Enterprise Manager* a konzoli *Query Analyzer* ve verzi 2000, *SSIS Package Utility* (`dtutil`) a *SSIS Package Execution Utilit* (`dtexec` a `dtexecui`).

5. 4 SSIS v praxi

Tolik tedy o Integration Services teoreticky. Nejlépe se ovšem slabiny a přednosti nástroje ukáží až při realizaci nějakého projektu. V této kapitole bych rád předvedl část ETL pro data Intrastatu z minulé kapitoly, ovšem nikoli v DTS, ale právě v SSIS.

Ačkoli byl scénář poněkud zjednodušen[43], v podstatných rysech odpovídá realitě. V SQL Serveru 2005 jsem vytvořil 3 databáze „Istat_zdroj“, „Istat_stage“ a „Istat_dw“. Jejich účel je jasný z názvů. Databáze „Istat_zdroj“ obsahuje zdrojové tabulky „Intrastat“, „Cis_Address“ a „Cis_Kodpohybu“. V „Istat_stage“, představující Stage oblast, jsou tabulky „Cis_Address“ a

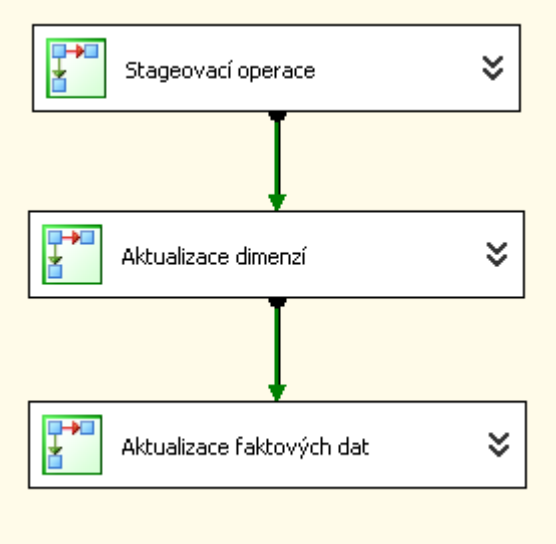
[41] Pomocí nastavení souboru boot.ini ji lze zvýšit na 3 GB [21].

[42] Management Studio poskytuje stejné funkce jako Enterprise Manager, např. grafické zobrazení balíčků, přidány jsou možnosti sledování a řízení běhu balíčků.

[43] Až na časovou nejsou do řešení zahrnuty dimenze, které se nenačítají s daty Intrastatu, což s sebou přináší i snížení počtu sloupců faktové tabulky a menší počet lookupů. Také objem dat ve faktové tabulce byl z výkonnostních důvodů podstatně snížen.

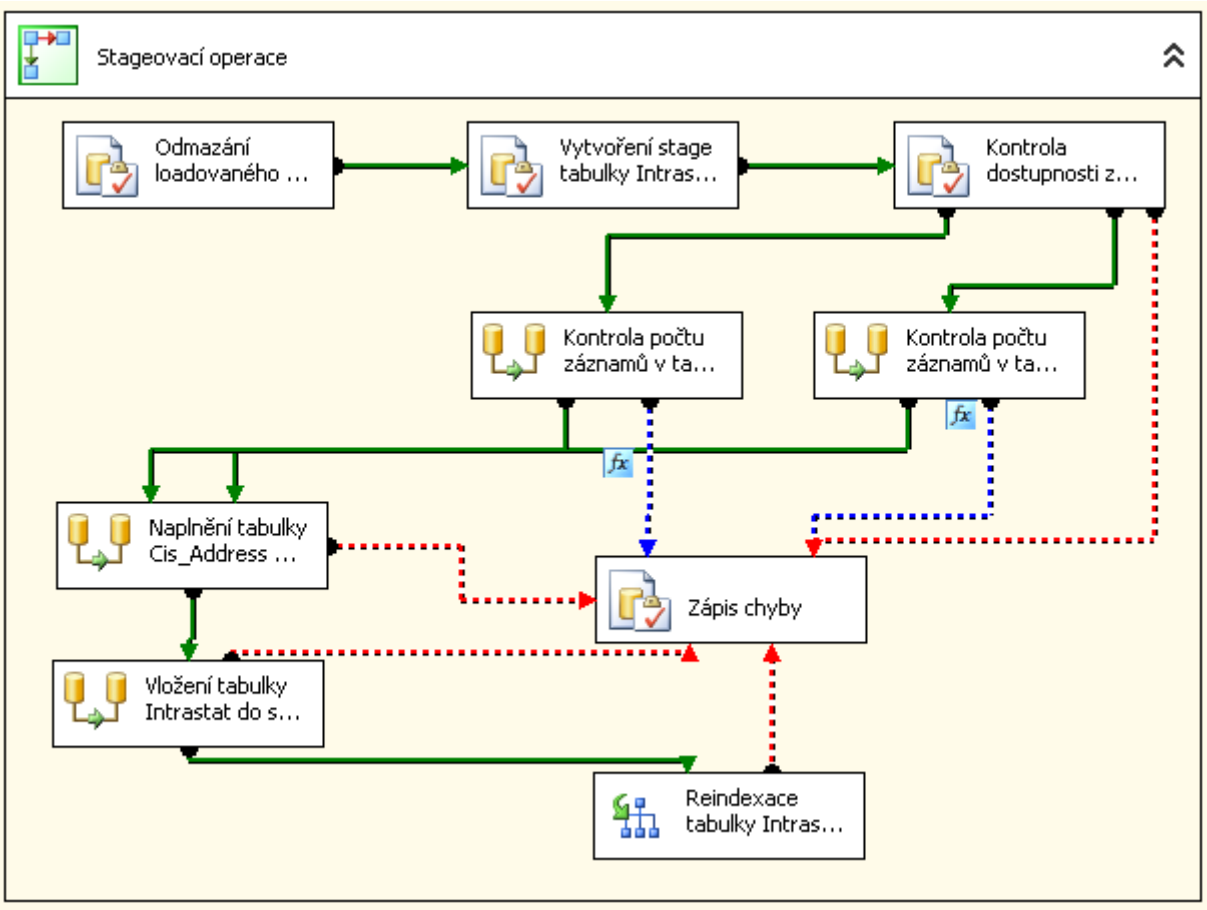
„Intrastat“. Datový sklad „Istat_dw“ se skládá z faktové tabulky „TFIntrastat“ a dimenzionálních tabulek „TDSubjekt“, „TDTypVetyIstat“ a „TDCasova“ . Do tabulky „Intrastat“ v databázi „Istat_zdroj“ byl načten vzorek dat, představující asi 13 000 záznamů. Zdrojové číselníky obsahují reálný objem dat a do časové dimenze byly vygenerovány data pro rok 2006.

Základní SSIS balíček se jmenuje „IntrastatLoad“ a tvoří ho tři kontejnery: „Stageovací operace“, „Aktualizace dimenzí“ a „Aktualizace faktových dat“.



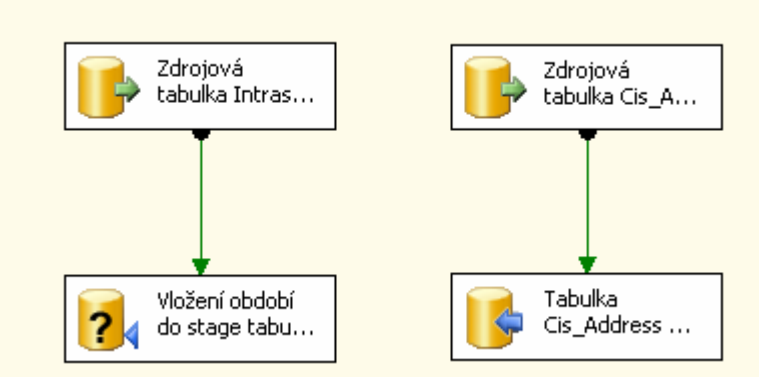
Obr. 26 - Balíček "IntrastatLoad"

5. 4. 1 Načtení dat do Stage



Obr. 27 - Balíček "Stageovací operace"

Začátek je podobný jako u DTS řešení. Po odmazání načítaného období z „TFIntrastat“ v datovém skladu je znovuvytvořena tabulka „Intrastat“ ve Stage a jsou provedeny kontroly dostupnosti zdroje a existence záznamů ve zdrojových tabulkách. Poté je ze zdrojové tabulky „Cis_Address“ naplněna odpovídající tabulka ve Stage. Další zdrojový číselník „Cis_KodPohybu“ není do Stage vůbec načítán, protože obsahuje malé[44] množství dat a lze u něj využít výhodu SSIS, totiž zpracování dat v paměti[45]. V dalším kroku je do Stage načtena tabulka „Intrastat“. Samotné vkládání probíhá probíhá pomocí Data Flow úlohy OLE DB Command[46].



Obr. 28 - Vložení tabulky „Intrastat“ a „Cis_Address“ do Stage

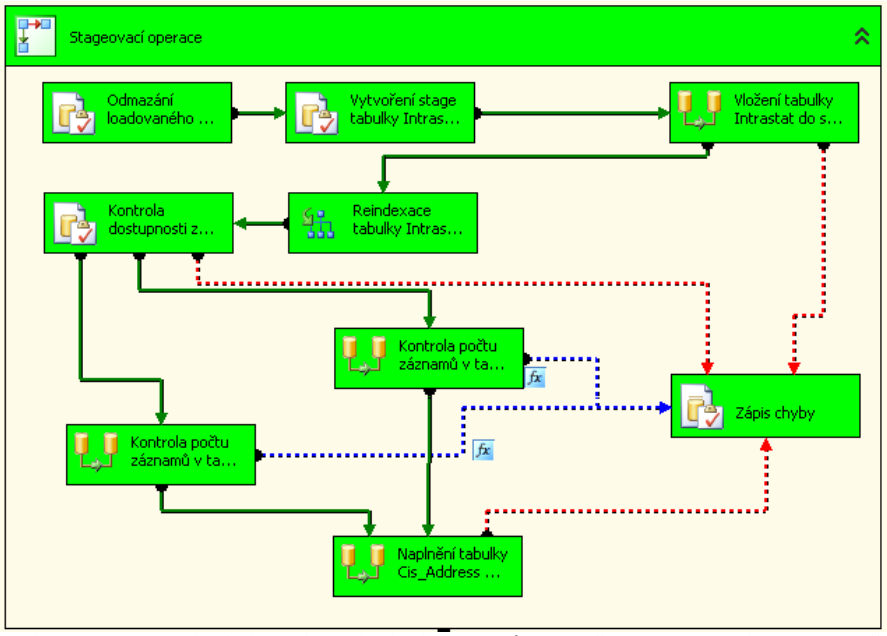
[44] V době psaní práce se jednalo pouze o 12 záznamů.

[45] Tento postup jsem zvolil pouze abych demonstroval tuto výhodu SSIS. V reálu by stage použita být měla, aby mohlo dojít k rychlému načtení dat a zbytek ETL procesu tak mohl být na zdroji dále nezávislý.

[46] Pokud by byla zdrojová data i stage na SQL Serveru 2005, mohlo by být efektivnější použít jako stage Raw File uložiště.

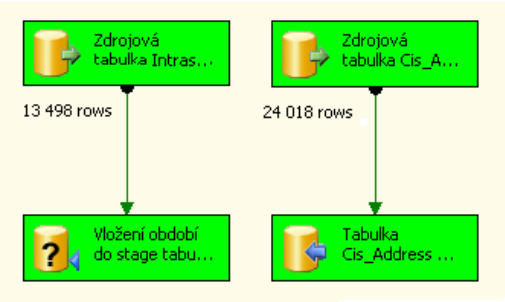
Nakonec je pomocí předpřipravené Control Flow úlohy provedena reindexace. Pokud některá z operací skončí chybou nebo pokud budou výsledky kontrol nepříznivé, dojde k zápisu do logovací tabulky.

Po spuštění vykonávání projektu (ladění) je u každého kontejneru, balíčku a úlohy vidět, jakým výsledkem jeho vykonávání skončilo. Pokud je zbarven žlutě, znamená to, že vykonávání probíhá a bílá znamená, že (zatím) není vykonáván. Kontejner/balíček/úloha může skončit v jednom ze dvou stavů – úspěšné dokončení (zelená) nebo selhání (červená). Pokud selže kterákoli z úloh v kontejneru, červený bude i celý kontejner.



Obr. 29 - Výsledek vykonávání kontejneru "Stageovací operace"

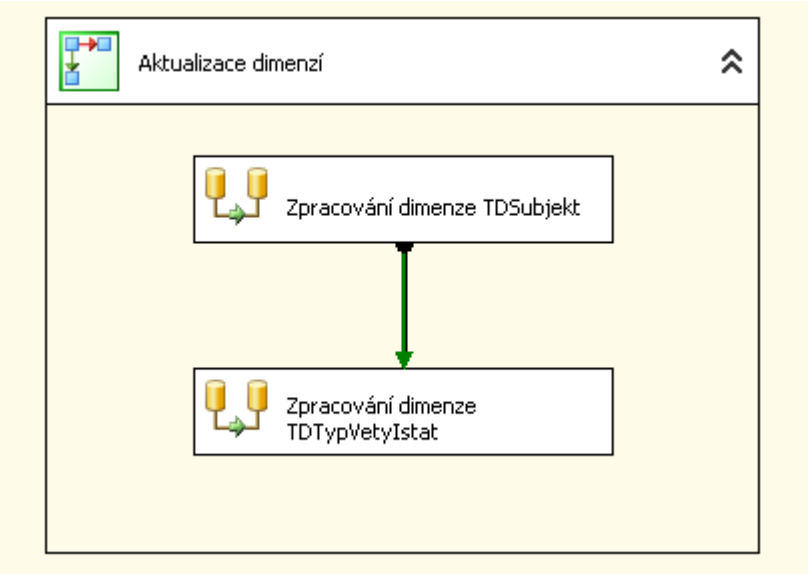
Úlohy typu Data Flow signalizují nejen stav, ale i počet přenesených či upravených záznamů v jednotlivých větvích diagramu.



Obr. 30 - Výsledek vkládání tabulek "Intrastat" a "Cis_Address" do Stage

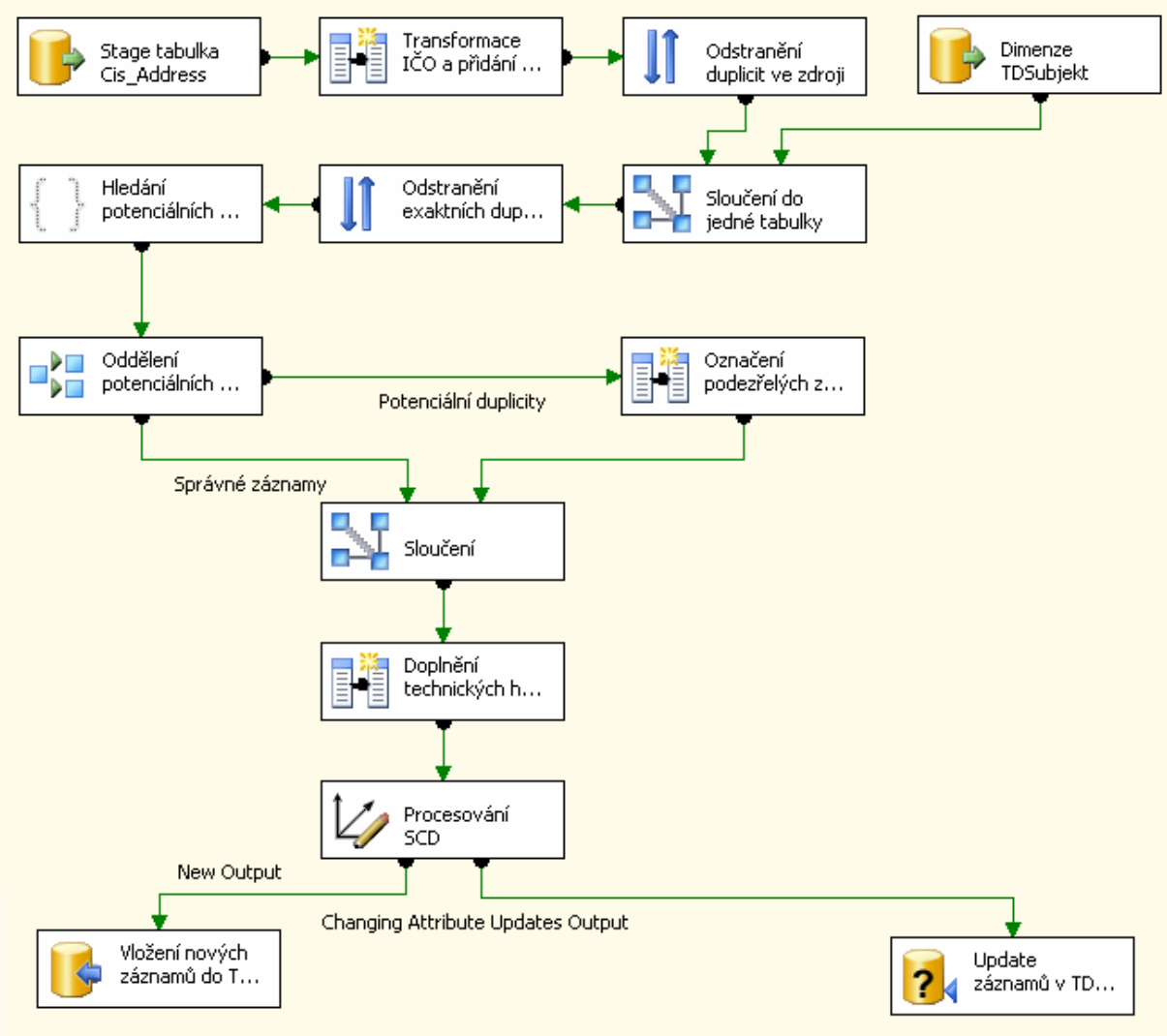
Po úspěšném dokončení kontejneru „Stageovací operace“ je spuštěn kontejner „Aktualizace dimenzí“.

5. 4. 2 Aktualizace dimenzí



Obr. 31 - Kontejner "Aktualizace dimenzí"

Tvoří ho pouze dvě Data Flow úlohy, které slouží k aktualizaci dimenze „TDSubjekt“ resp. „TDTypVetyIstat“. Záložka Data Flow první z nich obsahuje diagram zachycený na Obr. 30.



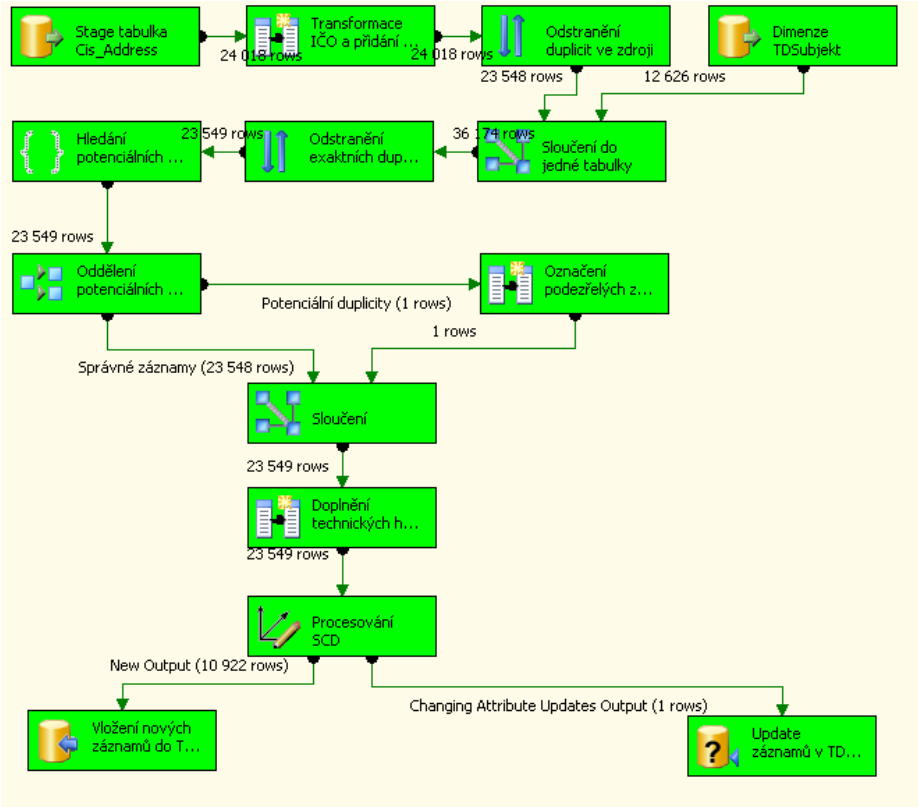
Obr. 32 – „Zpracování dimenze TDSubjekt“

K Stage tabulce „Cis_Address“ je, pro pozdější využití, pomocí úlohy Derived Column přidán sloupec „Podezreni“ a na požadovaný formát upraveno DIČ subjektu (před řetězec je přidáno `,CZ'`). Následuje úloha Sort, jenž neposkytuje pouze funkci třídění záznamů, ale pokud je zaškrtnuta volba *Remove rows with duplicate sort values*, lze z tabulky s její pomocí snadno odstranit duplicity. Zde je jako sloupec k třídění označen „DIČ“ (přirozený klíč tabulky), čímž je znemožněno načtení subjektů se shodným DIČ ze Stage (resp. ze zdroje). Úloha typu Union All sloučí data ze Stage s obsahem dimenze „TDSubjekt“ a poté jsou opět pomocí Sort úlohy vymazány záznamy u kterých se shoduje DIČ a název subjektu. Následuje úloha „Fuzzy Grouping“, která by se měla vypořádat i se záznamy, které sice nejsou úplně shodné, ale představují potenciální duplicity[47]. Parametr Similarity (míra podobnosti) byl u DIČ a telefonního čísla nastaven na 0,9. To znamená, že aby byl záznam označen za potenciální duplicitu, musel by se z 90% shodovat s jiným. U názvu firmy a e-mailové adresy byla míra shody určena na 70%. U podezřelých záznamů je do sloupce „Podezreni“ doplněno „`Podezření - duplicitní záznam`“[48]. Poté jsou větve s podezřelými i validními záznamy opět sloučeny a následuje doplnění technických hodnot „`#Neuvedeno`“ místo prázdných řetězců nebo `NULL` hodnot a zápis o neúplném záznamu do sloupce „Podezreni“. Nakonec je k rozhodnutí, zda záznam upravit nebo vložit jako nový využita úloha Slowly

[47] Tímto by bylo možné rozpoznat např. překlepy nebo dva různé zápisy stejného názvu subjektu typu „Novák, s. r. o.“ a „Novák, spol. s r. o.“.

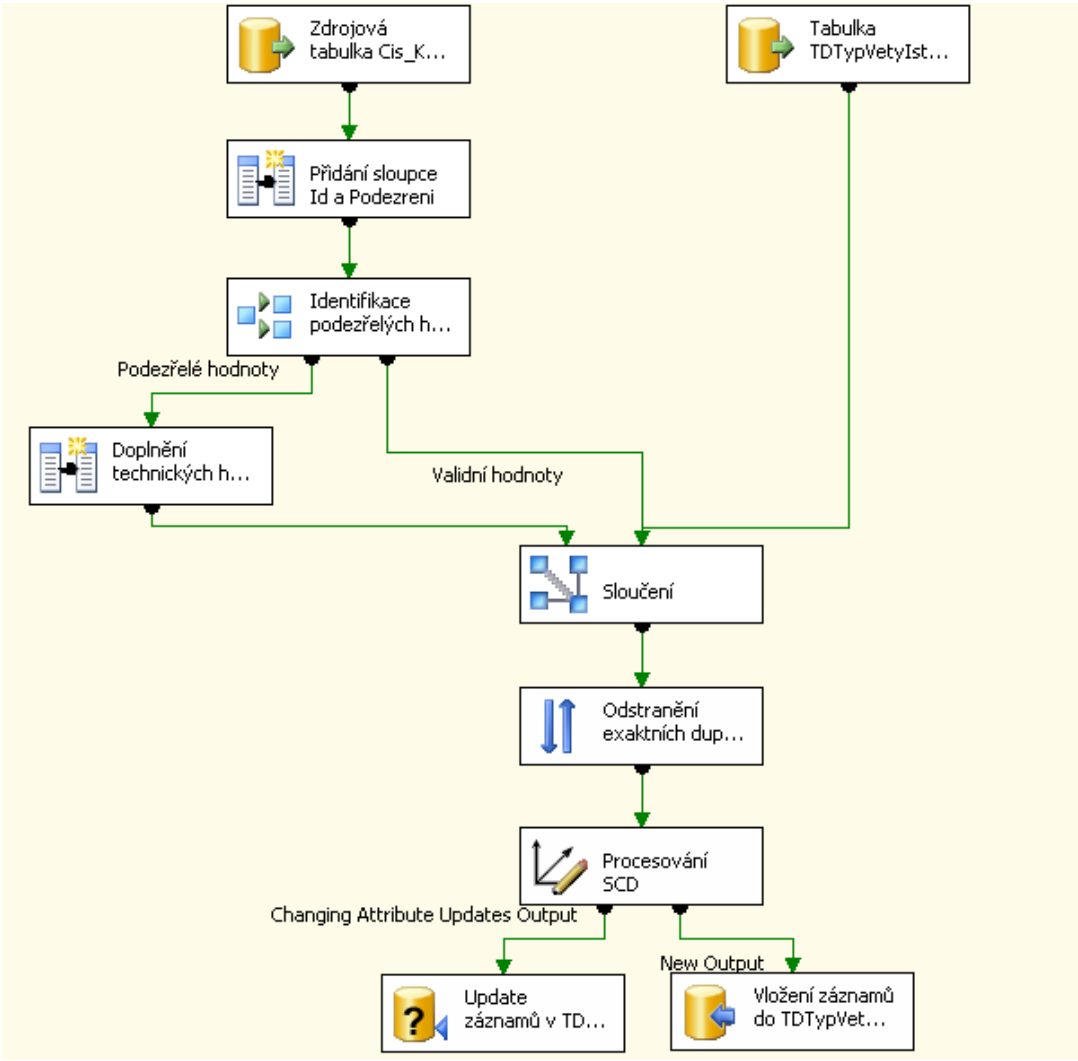
[48] Jedná se o zjednodušení, na reálném projektu by byl doplněn odkaz na klíč dimenze „TDPodezreni“, jak určují pravidla normalizace.

Changing Dimension. Business Key, jenž zde reprezentuje přirozený klíč dimenze, tvoří DIČ subjektu. Za Changing Attribute byly označeny sloupce „NazevSubjektu“, „telefon“ a „email“, které tedy budou aktualizovány podle zdroje. Nové záznamy (resp. záznamy s novým DIČ) budou do „TDSubjekt“ vloženy.



Obr. 33 - Výsledek „Zpracování dimenze TDSubjekt“

Po úspěšném dokončení činností v kontejneru je spuštěno vykonávání „Zpracování dimenze TDTypVetyIstat“.



Obr. 34 – „Zpracování dimenze TDTypVetyIstat"

Data pro aktualizaci dimenze jsou čerpána přímo ze zdrojové tabulky „Cis_Kodpohybu". K těm jsou, pro pozdější využití, přidány sloupce „Id" a „Podezreni" a jsou identifikovány podezřelé hodnoty. Dojde k rozdělení toku dat pomocí úlohy Conditional Split, obsahující následující podmínku:

```
kod == "" || text == "" || ISNULL(kod) || ISNULL(text)
```

Po identifikaci jsou `NULL` hodnoty a prázdné řetězce ve sloupcích „kod" a „text" nahrazeny technickými hodnotami „`#neznámo`" a „`#N`". To zajišťuje výraz v Derived Column úloze.

```
(ISNULL(text) || TRIM(text) == "") ? "#Neuvedeno" : text
```

Toky dat jsou posléze sloučeny a následuje odstranění duplicitních záznamů. Za duplicity jsou považovány řádky, u kterých se shodují sloupce "kod" a "text". Pak dochází k řešení SCD problému. Využit je znovu SCD Wizard, ve kterém je za Business Key označen sloupec "kod" a jako Changing Attribute sloupec "text", jenž obsahuje popis typu věty. Záznamy opět nejsou historizovány, nové jsou přidány a změny jsou řešeny přepsáním stávajících záznamů.

	kod	text
1	NULL	Podezřelá hodnota
2	MZ	Malé zásilky
3	NN	Negativní deklarace
4	ST	Běžný typ věty
5	ZI	Průmyslové celky
6	ZK	Kosmické lodi
7	ZL	Vlastnictví letadel
8	ZO	Odpady
9	ZP	Vlastnictví plavidel
10	ZR	Časově rozlišené zásilky
11	ZT	Technická zařízení na volném moři
12	ZV	Zboží pro vojenské účely
13	ZZ	Zásoby pro letadla a plavidla

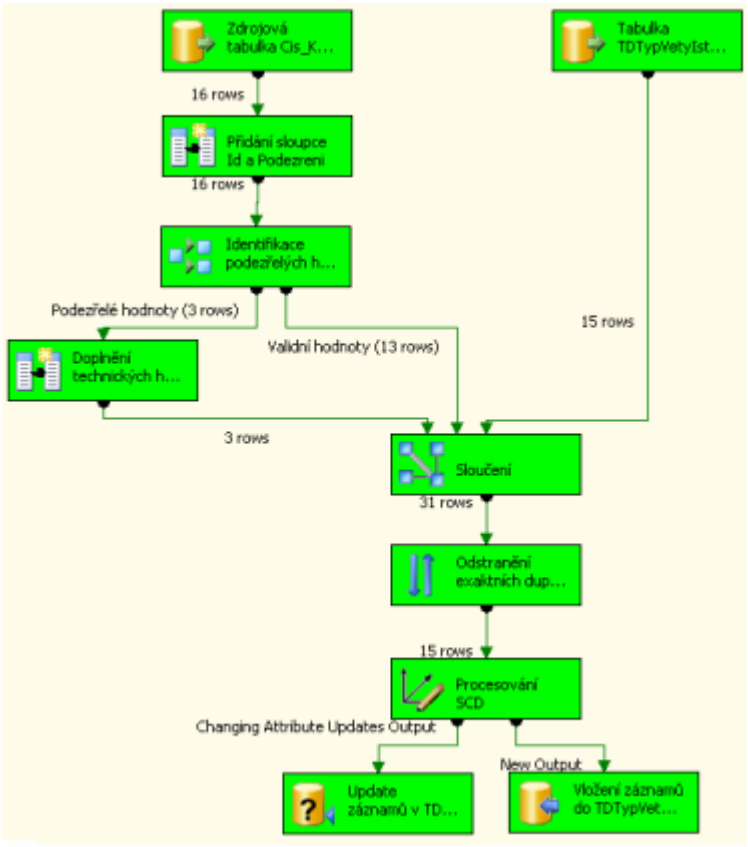
	TypVetyIstatId	KodTypuVety	PopisTypuVety	Podezreni
1	3	MZ	Malé zásilky	NULL
2	4	NN	Negativní deklarace	NULL
3	6	ST	Běžný typ věty	NULL
4	7	ZI	Průmyslové celky	NULL
5	9	ZL	Vlastnictví letadel	NULL
6	10	ZO	Odpady	NULL
7	11	ZP	Vlastnictví lodí	NULL
8	13	ZT	Technická zařízení na volném moři	NULL
9	14	ZV	Zboží pro vojenské účely	NULL
10	15	ZZ	Zásoby pro letadla a plavidla	NULL

Obr. 35 - Zdrojová data a původní data v "TDTypVetyIstat"

Obrázek ukazuje zdrojová data a původní stav tabulky „TDTypVetyIstat“. Následující obrázek pak provedené změny. Poslední tři záznamy byly vloženy nově, přičemž do prvního z nich byla doplněna technická hodnota a druh podezření. U jiného záznamu byl změněn popis z „`Vlastnictví lodí`“ na „`Vlastnictví plavidel`“.

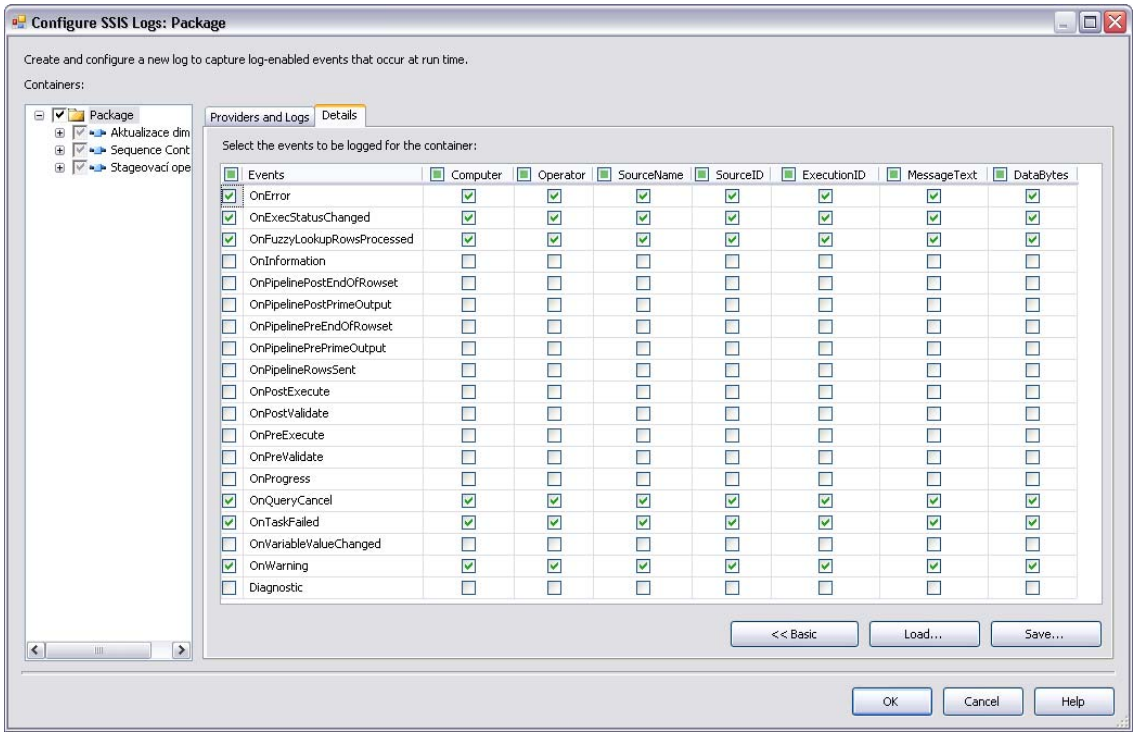
	TypVetyIstatId	KodTypuVety	PopisTypuVety	Podezreni
1	3	MZ	Malé zásilky	NULL
2	4	NN	Negativní deklarace	NULL
3	6	ST	Běžný typ věty	NULL
4	7	ZI	Průmyslové celky	NULL
5	9	ZL	Vlastnictví letadel	NULL
6	10	ZO	Odpady	NULL
7	11	ZP	Vlastnictví plavidel	NULL
8	13	ZT	Technická zařízení na volném moři	NULL
9	14	ZV	Zboží pro vojenské účely	NULL
10	15	ZZ	Zásoby pro letadla a plavidla	NULL
11	16	#N	Podezřelá hodnota	Podezření - nevyplněné hodnoty
12	17	ZK	Kosmické lodi	NULL
13	18	ZR	Časově rozlišené zásilky	NULL

Obr. 36 - Aktualizovaná data v "TDTypVetyIstat"



Obr. 37 - Výsledek „Zpracování dimenze TDTypVetyIstat"

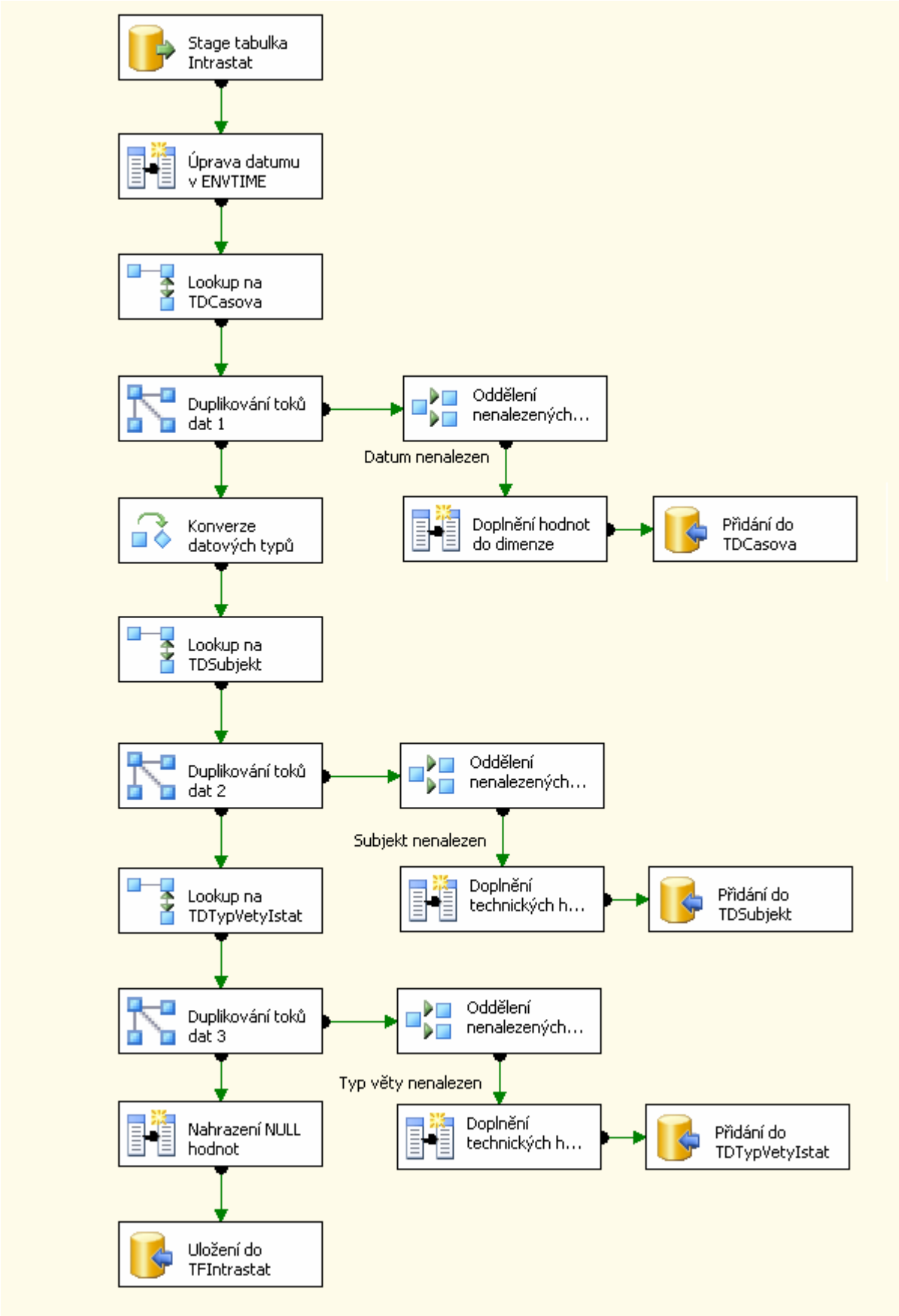
Narozdíl od prvního kontejneru, nejsou v “Aktualizace dimenzí” chyby zapisovány do tabulky “TDEtlLog”, ale využil jsem alternativu, kterou nabízí SSIS. Jedná se o zápis logovacích informací do speciální tabulky `sysdtslog90`. Stačí zde nastavit, které události se mají sledovat.



Obr. 38 - Nastavení logování

5. 4. 3 Aktualizace faktových dat

Poté, co je dokončena aktualizace dimenzí, může dojít k načtení dat do faktové tabulky “TFIntratat”. To zajišťuje kontejner “Aktualizace faktových dat”, který obsahuje jedinou Data Flow úlohu (“Zpracování TFIntrastat”). Po přepnutí na záložku Data Flow se objeví diagram, znázorněný na následujícím obrázku.

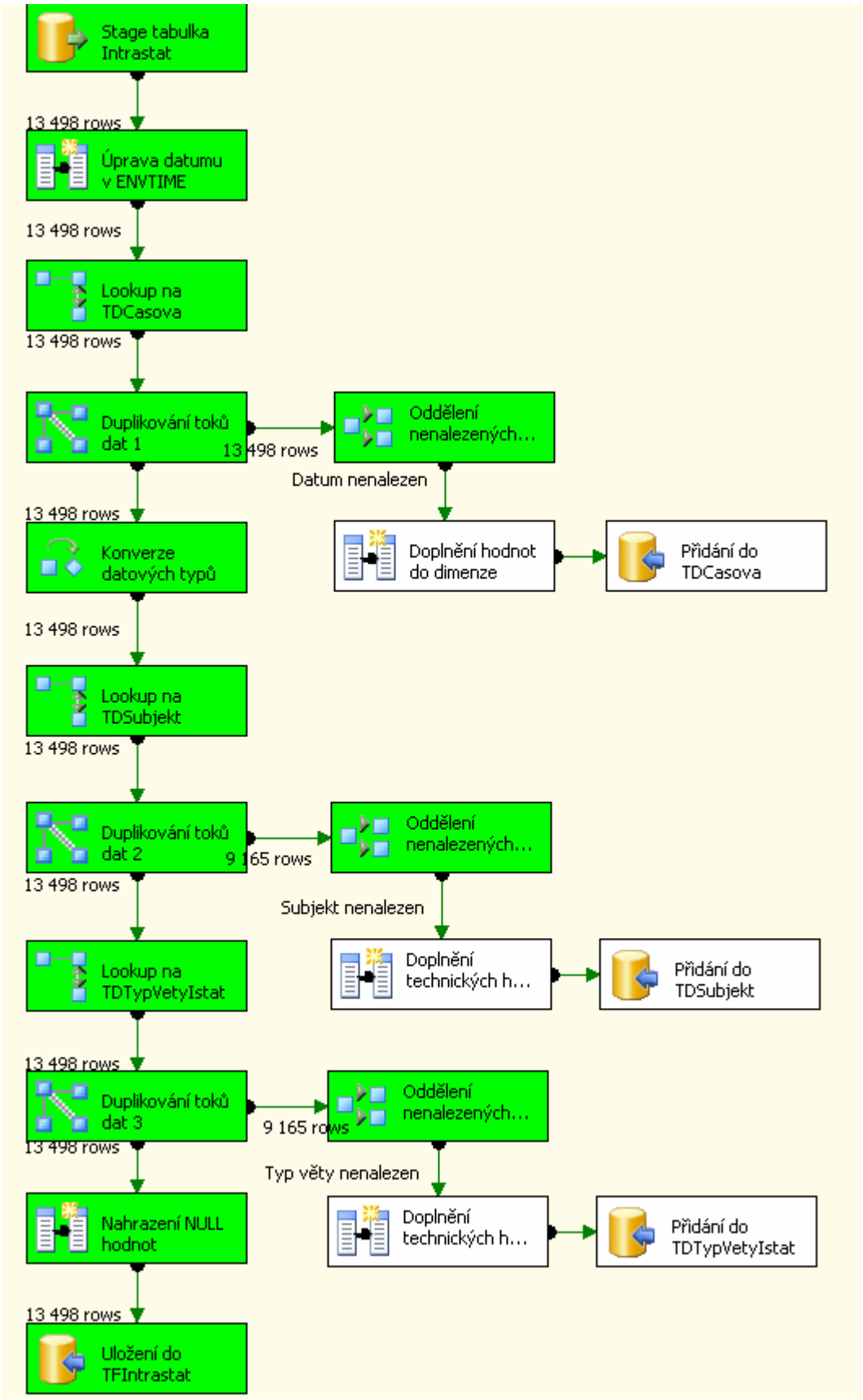


Obr. 39 - "Zpracování TFIntrastat"

Nejdříve jsou načtena data ze Stage a po úpravě sloupce „ENVTIME“, představující datum záznamu, je proveden lookup na časovou dimenzi. Místo datumu bude v tabulce „TFIntrastat“ figurovat odkaz na klíč dimenze „TDCasova“. Jestliže nebude datum v dimenzi nalezeno, bude v ní zaveden nový záznam, jehož hodnoty („Den“, „Mesic“, „Rok“, „Kvartal“, atd.) jsou získány z datumu v „ENVTIME“. To je umožněno zapnutím volby *Ignore Failure* v úloze Lookup. Pokud by zapnuta nebyla, po nenalezení datumu v dimenzi by lookup skončil chybou.

Po nezbytné konverzi datových typů dojde k lookupu na dimenzi “TDSubjekt”. Jako spojovací článek je zde použit přirozený klíč dimenze, kterým je “KodSubjektu”. Odpovídající sloupec “DIC” v tabulce “Intrastat” je nahrazen primárním klíčem “TDSubjekt”, což je “SubjektId”. Jestliže se v “DIČ” vyskytuje hodnota, kterou nelze v “TDSubjekt” dohledat, musí tam být opět doplněna. U subjektu ale nelze název firmy nebo telefon odvodit, proto jsou místo nich doplněny technické hodnoty “`#neznámo`“ a do sloupce „Podezreni“ vloženo „`Podezření – nenalezeno lookupem`“. Analogický postup je použit i u dimenze „TDTypVetyIstat“[49]. `JOIN` probíhá přes sloupec „KodTypuVety“ (na straně dimenzionální tabulky) a „TYPDEC“ u Stageové tabulky. Před načtením dat do faktové tabulky je ještě zkontrolováno, zda měrné jednotky (které zde představují sloupce „HMOT“, „MNOZMJ“ a „FAKHOD“ zdrojové tabulky) neobsahují `NULL` hodnoty a pakliže ano, je místo nich doplněna numerická nula.

[49] Reálná faktová tabulka obsahuje i odkazy na další dimenze, ale vzhledem k tomu, že se nenačítají s daty Intrastatu a princip lookupu je u nich naprosto stejný, dovolil jsem si úlohu zjednodušit a vynechat je.



Obr. 40 - Výsledek "Zpracování TFIntrastat"

Po úspěšné aktualizaci faktových dat by ještě měla následovat úprava tabulky „TQIntrastat“, což je postup opačný, než jsem výše prezentoval. Umělé klíče ve faktové tabulce jsou kvůli zvýšení dotazovacího výkonu nahrazovány přirozenými. Nakonec by měl být zapsán příznak pro zpracování OLAPu do tabulky „TSParametr“.

Na základě zkušeností s praktickým využitím SSIS jsem sestavil následující tabulku, která srovnává některé charakteristiky nástrojů. Jedná se o vlastnosti, které jsem považoval za důležité pro tvorbu ETL řešení.

	DTS	SSIS
Větvení	Poskytuje pouze větve OnCompletion, OnFailure a OnSuccess.	Robustní model, umožňující větvit podle různých událostí a/nebo na základě zadaných výrazů.
Skriptování	Podporuje VB Script, JScript, možnost doinstalace dalších.	Podpora VB.NET. Umožňuje vizuální návrh, předkompilaci, přidání breakpoints, atd.
Podpora transformací	Pomocí T-SQL a skriptování.	Nabízí široké spektrum úloh pro transformace a možnost tvorby T-SQL a VB.NET skriptů.
Použití Stage a dočasných tabulek.	Použití Stage je nutné. Během transformací se musí data ukládat do dočasných tabulek.	Využití Stage není nutné, ale správně navržené ETL by ji obsahovat mělo. Používat dočasné tabulky není potřeba, transformace se vykonávají s využitím virtuální paměti.
Logování	Vývojář musí pomocí skriptů vytvořit vlastní způsob logování.	Poskytuje několik variant automatického logování.
Reakce na chybové události	Po selhání se celý ETL proces vykonává od začátku.	Podpora checkpoints, umožňující pokračovat v procesu od místa selhání.
Škálovatelnost	Velmi omezená.	Podporována, škálovat lze např. přidělenou paměť.
Zajištění datové kvality	Neposkytuje.	Úlohy Sort a Fuzzy Grouping poskytují dobré možnosti pro vyčištění načítaných dat od duplicit.
Vývojové prostředí	Přehledné, ale manipulace s více otevřenými okny je uživatelsky nepříjemná.	Velmi intuitivní ovládání, dobré rozvržení plochy a rozmístění nástrojů. Samotný návrh spřehledňují kontejnery. Uživatelé si budou muset zvykat na oddělení datových toků od procesních.

Tabulka 3 - Porovnání některých vlastností nástrojů

5. 5 Srovnání DTS a SSIS

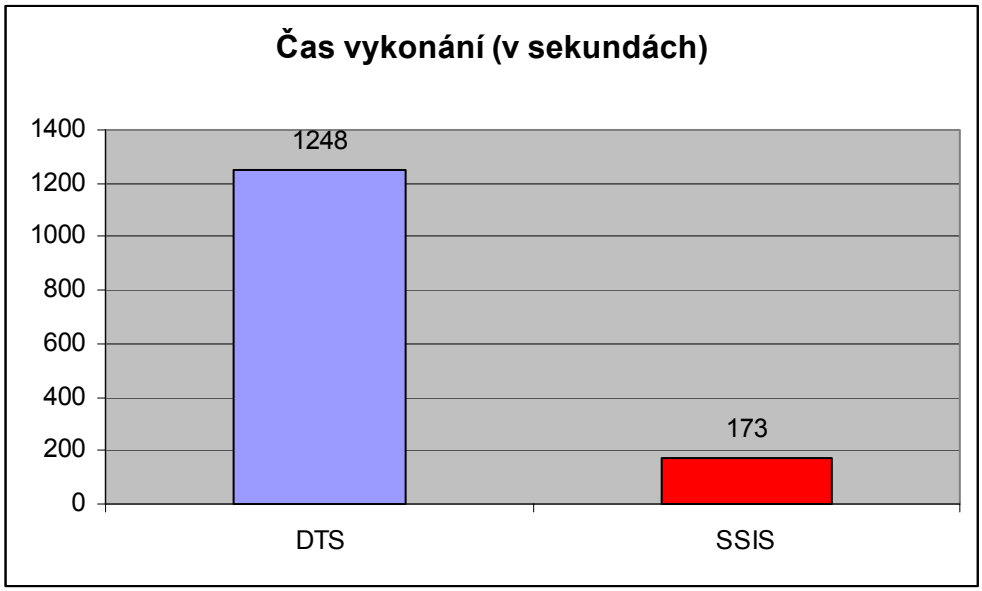
Nový nástroj neměl, vzhledem k oblíbenosti DTS, příliš snadnou startovní pozici na trhu. Velká část odborné veřejnosti zaujímala k SSIS od začátku poměrně skeptický postoj. Ačkoli nedostatky DTS byly obecně známé, vývojáři se naučili je obcházet a řešit, zejména pomocí psaní vlastního kódu a DTS byly (a stále ještě jsou) masivně využívány. Hlavním argumentem, který svědčil v neprospěch SSIS, byl fakt, že Microsoft hlásal, že přijde s úplně novým konceptem a nenechá „kámen na kameni". A proč úplně zavrhnout řešení, které sice „má své mouchy", ale jinak poměrně dobře a spolehlivě funguje?

Velké nevýhody ETL nástroje SQL Serveru 2000, jako jsou zejména obsluha chybových událostí a nedostatečná škálovatelnost, byly odstraněny. V Integration Services se workflow řídí nejen pomocí podmínek úspěšného či neúspěšného vykonání, ale je možné definovat vlastní podmíněné výrazy. Nový nástroj také poskytuje vývojářům možnost pružně reagovat nejen na chybové, ale i na široké spektrum jiných událostí. Navíc selhání již není nutné řešit pro každý balíček zvlášť. Pokud přičteme propracovanou možnost logování a přidání checkpoints, umožňující pokračovat v loadu až od místa přerušení, miska vah se začíná silně vychylovat ve prospěch Integration Services.

Zajímavým vylepšením je i teoretická možnost přestat používat Stage. Teoretická proto, že pro velké tabulky (např. zdrojové tabulky pro tabulky faktů, velké číselníky jako jsou zákazníci, zboží, atd.) je použití Stage v každém případě nezbytné. Nicméně vidina využití virtuální paměti, na úkor dočasných tabulek při transformaci dat, je reálná a lákavá. Další naprostou novinkou je Fuzzy logika, používaná pro rozpoznávání duplicitních záznamů. Dle mého názoru se jedná o velmi dobrého pomocníka, který by mohl najít uplatnění i na skutečně velkých projektech. Naproti tomu využitelnost Slowly Changing Dimensions Wizard na reálných projektech (zejména pro typy dimenzí, u kterých se historizuje) budí pochyby. Microsoft ale vyvinul chvályhodnou snahu nabídnout vývojářům možnost vyřešit donedávna složitý problém SCD, pomocí předdefinované úlohy, během několika málo minut. I kdyby tato úloha nebyla v praxi příliš často využívána, je možné, že v příští verzi ETL nástroje od Microsoftu dostanou ETL specialisté robustní nástroj pro řešení SCD.

Na oddělení kontrolních a datových toků si nejspíš budou muset dosavadní uživatelé DTS chvíli zvykat. Ale jsem toho názoru, že po nějaké době používání nástroje, dospějí k přesvědčení o logické správnosti takového řešení. Co se týče samotného návrhu, výborným nápadem bylo zahrnutí kontejnerů a možnost ladění pomocí breakpoints. Vývojové prostředí je velmi přehledné a celkově působí uživatelsky velice přívětivě. Snaží se vývojářům usnadnit práci tím, že pro drtivou většinu běžných operací nemusí psát skripty. Přestože je ale psaní kódu sice mnohdy zdlouhavé a nepohodlné, jeho autor přesně ví, jaké činnosti provádí a snadněji v něm najde chybu. Proto je (alespoň dle mého názoru) nejlepší kompromis, to znamená pro rutinní operace používat předpřipravené nástroje, pro složitější činnosti psát vlastní skripty. A k tomu poskytují Integration Services dostatečnou výbavu.

Aby se moje hodnocení neskládalo pouze z kvalitativních ukazatelů, uvedu zde graf, který porovnává čas vykonání (v sekundách) stejných úloh v DTS a SSIS. Integration Services z tohoto souboje vyšly jako jednoznačný vítěz.



Obr. 41 - Porovnání výkonu DTS a SSIS. Zdroj: [24]

5. 5. 1 Praktické zkušenosti

Zajímavé porovnání obou produktů nabídl mnou prezentovaný „reálný“ příklad. Načítání do Stage bylo až na drobnosti podobné, s tím rozdílem, že v Integration Services bylo nutné načíst pouze faktová data a číselník subjektů.

Velké odlišnosti se ale objevily v aktualizaci dimenzí. Zatímco na projektu byl použit velký počet uložených procedur, jejichž vývoj byl jistě časově náročný, v SSIS bylo možné využít širokou škálu předpřipravených úloh, což práci značně usnadnilo. Vyzdvihl bych zejména odstranění duplicitních hodnot, pro které bylo původně nutné vytvořit a porovnávat dvě dočasné tabulky. Integration Services ale umožňují nalezení shodných záznamů pomocí jediné úlohy (Sort). Zajímavá je možnost identifikace pravděpodobných duplicit. Ačkoli je proces odhalování poněkud pomalý (protože Integration Services zde používají speciální indexování), může být velmi užitečné úlohu do projektu zahrnout a zajistit větší datovou kvalitu, aniž by firma investovala obrovské peníze do speciálních nástrojů. Oproti očekávání se mi velmi osvědčilo řešení SCD pomocí úlohy. Pokud se do dimenze pouze přidávají nové, mění stávající záznamy a nehistorizuje se, jde o velmi dobrý způsob, jak problematiku SCD za krátký čas zvládnout.

Ačkoli řešení aktualizace dimenzí bylo v SSIS jednoznačně lepší, a to jak z hlediska pohodlí návrháře ETL, tak z hlediska výkonu, u faktů tomu tak, alespoň co se výkonu týče, nebylo. Návrh provedení lookupu a identifikace nových záznamů sice byl díky úlohám relativně jednoduchý, ale celý proces aktualizace faktových dat se neúměrně prodloužil. Proto by bylo vhodnější pro fakta využít stávající řešení pomocí uložené procedury.

5. 5. 2 Srovnání s ostatními ETL nástroji

A jak by dopadlo srovnání s ostatními ETL nástroji na trhu? Kvalifikovaná odpověď by vyžadovala hlubší analýzu, ale troufal bych si tvrdit, že v konkurenci nástrojů od ostatních databázových výrobců (Oracle, IBM), se SSIS prosadí. Ačkoli je Microsoft na poli transakčních databází za Oraclem stále o „nějaký ten krůček“ pozadu, jeho ETL nástroj nabízí komplexnější výbavu pro transformace, větvení, logování, obsluhu událostí, škálovatelnost a paralelizaci než *Oracle Warehouse Builder*. V porovnání se specializovanými ETL nástroji, jako jsou např. *Ab Initio, Informatica PowerMart* a podobnými, se situace zřejmě obrátí v neprospěch SSIS. Na druhou stranu, srovnají-li se ceny, které jsou u těchto ETL nástrojů

několikanásobně vyšší a vezme-li se v potaz fakt, že Integration Services zákazník koupí v jednom balíku s transakční databází, Analysis a Reporting Services, misky vah se minimálně vyrovnají.

6 ZÁVĚR

Cílem této bakalářské práce bylo porovnat Microsoft SQL Server Integration Services a Data Transformation Services z hlediska využitelnosti v etapě Extraction Transformation Loading budování datových skladů.

Od problematiky růstu objemu a významu dat v podnicích, přes vznik databází, má práce dospěla k Business Intelligence, datovým skladům a ETL, kde zmíněné nástroje nacházejí své využití. Poté došlo nejen k srovnání DTS a SSIS, ale i k praktickým ukázkám.

Pokud bych měl vše podtrhnout a sečíst, hypotéza, že Integration Services jsou lepším ETL nástrojem než Data Transformation Services, by byla potvrzena. Na trh byl uveden produkt, který vychází z DTS, zdědil jeho silné stránky a ze slabých se poučil. Většina neduhů DTS, na které si vývojáři stěžovali, je zapomenuta. SSIS nabízí pružné reakce na různé druhy událostí, škálovatelnost, ošetření chyb, možnost ladění, atd. K tomu navíc přidává „bonusové“ transformační úlohy, jako je řešení SCD, Fuzzy logiku a jiné. Praktická část mé práce ukázala, jak mohou být SSIS přínosné. Zejména aktualizace dimenzí by pomocí SSIS byla podstatně jednodušší než původní řešení.

Zkrátka Microsoft udělal na poli ETL nástrojů a nástrojů určených k datové integraci velký skok a uvedl na trh produkt, který je schopen DTS nejen plnohodnotně nahradit, ale který ho v mnoha ohledech předčí.

7 POUŽITÁ LITERATURA

[1] *Adastra Corporation* [web site]. Dostupné z: http://www.adastra.cz
Webové sídlo mezinárodní konzultační společnosti, která se věnuje především Data Warehousingu a Business Intelligence, Master Data Managementu, aplikačnímu vývoji Quality Assurance a outsourcingu. Stránka obsahuje informace o společnosti, nabízená řešení, reference, nabídku školení, atd.

[2] *Adastra prezentuje výsledky exkluzivního průzkumu o kvalitě dat v českých firmách*. [www dokument]. Dostupný z: http://www.adastra.cz/dokument.aspx?id=67
[cit. 3. 11. 2006]

[3] BRAY, Tim, PAOLI, Jean, SPERBERG-McQUEEN, C.M., MALER, Eve. *Extensible Markup Language (XML) 1.0 (Second Edition)* [www dokument], *W3C Recommendation 6 October 2000*. Dostupný z: http://www.w3.org/TR/REC-xml [cit. 24. 10. 2006]

[4] *Building Distributed Applications: Scaling Out SQL Server 2005. [Vývoj distribuovaných aplikací: Škálovatelnost SQL Serveru 2005]*. Microsoft Corporation [www dokument] 4/2006. Dostupný z: http://msdn2.microsoft.com/en-us/library/aa479364.aspx
[cit. 16. 12. 2006]

[5] COVENEY, Peter, HIGHFIELD, Roger. *Mezi chaosem a řádem. Hranice komplexity: hledání řádu v chaotickém světě*. Praha: Mladá Fronta, 2003, 432 s., ISBN: 80-204-0989-0

[6] ČERNOHORSKÝ, Petr. Řízení BI v mezinárodní společnosti. *Konference Business Intelligence*. 25. 4. 2006

[7] *Data Warehousing od A do Z*. Odborné školení. Materiál společnosti Adastra Corporation. 5. – 6. 10. 2006

[8] DE MONTCHEUIL, Yves, DUPUPET, Chris, Juha. *Third Generation ETL: Delivering best performance [Třetí generace ETL: Dodání nejvyššího výkonu]*. Sunopsis, White paper, 9 s. Dostupný z: http://www.tdwi.org/Marketplace/Whitepaper.aspx?pID=158
[dokument ve formátu PDF]

[9] FAYAAD, Usama M., PIATETSKY-SHAPIRO, Gregory, SMYTH, P., UTHURUSAMY, R. *Advances in Knowledge Discovery and Data Mining [Kroky v objevování znalostí v databázích a dolování dat]* MIT Press, 1998

[10] FIELDING, R. L. Business Intelligence Advancements Transform Corporate Decision-Making [Postupy Business Intelligence mění podnikové rozhodování]. *Business Intelligence.com*. [www dokument]. Dostupný z:
http://www.businessintelligence.com/ex/asp/code.161/xe/article.htm [cit. 20. 10. 2006]

[11] FRIEDMAN, T. *ETL Magic Quadrant Update: A Market in Evolution [Změna magického ETL kvadrantu: trh ve vývoji]*. Gartner Group, Inc. 6. 5. 2002 [www dokument]. Dostupný z: http://www.gartner.com/reprints/informatica/106602.html [cit. 10. 11. 2006]

[12] FROULÍK, Radek. Nová ekonomika a globální informační společnost. *Interval.cz: webdesigna e-komerce denně* [online]. 4. 5. 2005. Dostupný z:
http://interval.cz/clanky/nova-ekonomika-a-globalni-informacni-spolecnost/
[cit. 12. 10. 2006]

[13] GAVENDE, Sandesh. What is ETL? [Co je ETL?]. *ETL Guru: ETL Strategy for the Enterprise*. 24. 4. 2006.[www dokument]. Dostupný z: http://etlguru.com/blog/category/etl-basics/ [cit. 3. 11. 2006]

[14] HABÁŇ, Jaromír <haban@fame.utb.cz>, SODOMKA, Petr <sodomka@fame.utb.cz>. *Efektivní tvorba a provoz datových skladů*. Zlín Centrum pro výzkum informačních systémů, UTB, Fakulta managementu a ekonomiky, Ústav managementu výroby – průmyslové inženýrství, 7 s. Dostupný z: http://si.vse.cz/archiv/clanky/2003/sodomka.pdf [dokument ve formátu PDF]

[15] HATHI, Kamal. *An Introduction to SQL Server 2005 Integration Services. [Úvod do integračních služeb SQL Serveru 2005]* [www dokument] 1. 5. 2005. Dostupný z: http://www.microsoft.com/technet/prodtechnol/sql/2005/intro2is.mspx [cit. 1. 12. 2006]

[16] HUMPHRIES, Mark, HAWKINS, Michael W., DY, Michelle C. *Data Warehousing: Návrh a implementace*. Praha: Computer Press, 2002, 257 s., ISBN: 80-7226-560-1

[17] Chapter 22 – Cubes in the Real World [Kapitola 22 – Kostky v reálném světě] SQL Server 2000 Resource Kit [Zdrojová sada SQL Server 2000]. [www dokument]. Dostupný z: http://www.microsoft.com/technet/prodtechnol/sql/2000/reskit/part6/c2261.mspx?mfr=true [cit. 24. 10. 2006]

[18] INMON, William H. Building the Data Warehouse 3 rd Edition [Budování datových skladů 3. vydání]. New York: Wiley, 2002, 432 s., ISBN: 0471081302

[19] INMON, William H., The Operational Data Store: Designing the Operational Data Store [Operativní úložiště dat: Návrh Operativního úložiště dat]. *DM Review*. 1998. [www dokument]. Dostupný z: http://www.dmreview.com/article_sub.cfm?articleId=469 [cit. 23. 10. 2006]

[20] KIMBALL, Ralph, ROSS, Margy, MERZ, Richard. Data Warehouse Toolkit 2nd Edition: The Complete Guide to Dimensional Modeling [Sada nástrojů pro Datové sklady 2. vydání: Kompletní průvodce dimenzionálním modelováním]. New York: Wiley, 2002, 464 s., ISBN: 0471200247

[21] KNIGHT, Brian; MITCHELL, Allan; GREEN, Darren; HINSON, Douglas; KELLENBERGER, Kathi; LEONARD, Andy; VEERMAN, Erik; GERARD, Jason; JI, Haidong. *Professional SQL Server 2005 Integration Services [Profesionálně s SQL Server 2005 Integration Services]*. Wrox, 2006, 692 s., ISBN: 0764584359

[22] KUČERA, Milan. Dva způsoby budování datového skladu: Srovnání různých přístupů budování Data warehouse z pohledu investic. *IT systems*. 2003, Příloha 5/2001: Data warehousing a Business Inteligence. [www dokument]. Dostupný z: http://www.systemonline.cz/site/data-warehousing/archit2.htm [cit. 25. 10. 2006]

[23] KUČEROVÁ, Helena. *Databázové systémy : Sylaby ke kurzu*. Praha : Vyšší odborná škola informačních služeb, 2004. 110 s.
Dostupný z: http://info.sks.cz/users/ku/DOKUMENTY/das_syl.pdf
[dokument ve formátu PDF]

[24] LACKO, Ľuboslav. *Business Intelligence v SQL Serveru 2005: Reportovací, analytické a další datové služby*. Brno: Computer Press, 2006, 391 s., ISBN: 80-251-1110-5

[25] LACKO, Ľuboslav. Databáze: datové sklady, OLAP a dolování dat s příklady v Microsoft SQL Serveru a Oracle. Brno: Computer Press, 2003, 486 s., ISBN: 80-7226-696-0

[26] LÖFFELMANN, Jiří. Data warehousing a Business Inteligence. *IT systems*. 2002, Příloha 6/2002: Data warehousing a Business Intelligence. [www dokument]. Dostupný z: http://www.systemonline.cz/site/data-warehousing/upraven.htm [cit. 12. 10. 2006]

[27] McDOWELL, Douglas.*SQL Server 2005's ETL Tool: Integration Services. [ETL nástroj SQL Serveru 2005: Integrační služby]* [www dokument]. Dostupný z: http://www.sqlmag.com/Article/ArticleID/45910/SQL_Server_2005s_ETL_Tool_Integration_Services.html [cit. 30. 11. 2006]

[28] NOVOTNÝ, Ota, POUR, Jan, SLÁNSKÝ, David. *Architektury Business Intelligence*. Business Intelligence Magazine. Praha: Adastra Corporation. č. 2, červenec 2005.

[29] NOVOTNÝ, Ota, POUR, Jan, SLÁNSKÝ, David. *Business Intelligence: Jak využít bohatství ve vašich datech*. Praha: Grada Publishing, a. s., 2005, 256 s., ISBN: 80-247- 1094-3

[30] NOVOTNÝ, Ota, POUR, Jan, SLÁNSKÝ, David. *Business Intelligence v řízení firmy*. Business Intelligence Magazine. Praha: Adastra Corporation. č. 1, duben 2005.

[31] Oracle® Objects for OLE C++ Class Library Developer's Guide Release 9.2.0.4. *Manuál společnosti Oracle* [www dokument]. Dostupný z: http://download-east.oracle.com/docs/html/B10954_01/o4c00018.htm [cit. 16. 10. 2006]

[32] OTEY, Michael. *Managing and Deploying SQL Server Integration Services. [Řízení a přesuny v Integračních službách SQL Serveru]* [www dokument] 30. 4. 2005. Dostupný z: http://www.microsoft.com/technet/prodtechnol/sql/2005/mgngssis.mspx [cit. 30. 11. 2006]

[33] PAZDZIORA, Jan <adelton@fi.muni.cz>. Transakce v databázových serverech (a co s těmi, které transakce nemají). *Linuxové noviny*. 1999, č. 11. [www dokument]. Dostupný z: http://www.linux.cz/noviny/1999-11/clanek12.html [cit. 16. 10. 2006]

[34] PENDSE, Nigel <*nigelp@olapreport.com*>. *What is OLAP? An analysis of what the often misused OLAP term to mean. [Co je OLAP? Analýza co by měl často nesprávně používaný termín OLAP znamenat]* [www dokument] 15. 8. 2005. Dostupný z: http://www.olapreport.com/fasmi.htm [cit. 27. 10. 2006]

[35] POCHYLA, Martin <martin.pochyla@vsb.cz>. *Cesta k Business Intelligence*. Ostrava : VŠB-TU Ostrava, Ekonomická fakulta, Katedra informatiky v ekonomice, 8 s. Dostupný z: http://honor.fi.muni.cz/tsw/2001/153.pdf [dokument ve formátu PDF]

[36] POKORNÝ, Martin <redakce@dbsvet.cz>. Vyvíjíme databázový a informační systém IV. *Databázový svět* [online]. 26. 5. 2004. Dostupný z: http://www.dbsvet.cz/view.php?cisloclanku=2004052601 [cit. 13. 10. 2006]

[37] POKORNÝ, Martin <redakce@dbsvet.cz>. Vyvíjíme databázový a informační systém VII. *Databázový svět* [online]. 16. 6. 2004. Dostupný z: http://www.dbsvet.cz/view.php?cisloclanku=2004061601 [cit. 17. 10. 2006]

[38] POLÁŠEK, Marek. Databáze z hlediska podnikových informačních systémů. *IT systems*. 2003, Příloha 7-8/2003: Data warehousing a Business Inteligence. [www dokument]. Dostupný z: http://www.systemonline.cz/site/data-warehousing/7lofelmann.htm [cit. 13. 10. 2006]

[39] PŮLPÁN, Jaroslav. Dolování dat aneb hledání skrytých souvislostí. *IT systems*. 2001, Příloha 2001: Data warehousing a Business Inteligence. [www dokument]. Dostupný z: http://www.systemonline.cz/site/data-warehousing/datamine.htm [cit. 24. 10. 2006]

[40] RUD, Olivia Parr. Data mining: praktický průvodce dolováním dat pro efektivní prodej, cílený marketing a podporu zákazníků (CRM). Brno: Computer Press, 2001, 329 s., ISBN: 80-7226-577-6

[41] RYDVAL, Slávek <slavek@rydval.cz>. Normální formy. [online]. 7. 8. 2005. Dostupný z: http://www.rydval.cz/phprs/view.php?cisloclanku=2005123127 [cit. 17. 10. 2006]

[42] Řešení Data Warehouse a Business Intelligence (DW&BI) v prostředí Celní správy ČR. *Katalog obchodních řešení, průmyslových řešení a služeb* [www dokument]. Dostupný z: http://www.katalogreseni.cz/IndShowPrip.aspx?Show_id=prip&PripadovaStudie_id=272&returnTo=%2FIndSezPrip.aspx%3FShow_id%3Dprip [cit. 15. 11. 2006]

[43] SHARMA, Rahul, STEARNS, Beth, NG, Tony. *J2EE Connector Architecture and Enterprise Application. [Architektura připojení a podnikových aplikací v J2EE]* Boston: Addison Wesley Professional, 2002, 416 s., ISBN: 0-201-77580-8

[44] Schema Modeling Techniques [Techniky modelování schémat] *Oracle9i Data Warehousing Guide* [www dokument]. Dostupný z: http://www.lc.leidenuniv.nl/awcourse/oracle/server.920/a96520/schemas.htm#12915 [cit. 24. 10. 2006]

[45] SCHILLER, Martin. Co se skrývá pod zkratkou ETL? Jak zpracovat informace uložené v různých podnikových systémech. *IT systems*. 2003, Příloha 3/2003: Data warehousing a Business Inteligence. [www dokument]. Dostupný z: http://www.systemonline.cz/site/data-warehousing/etl3.htm [cit. 2. 11. 2006]

[46] SMÍŠEK, Jan <smisekj@iname.cz>. Globalizace a informační společnost. *Ne noviny* [online]. 16. 12. 1999, č. 3. Dostupný z: http://www.mujweb.atlas.cz/www/nenoviny/003/ginfspol.htm [cit. 12. 10. 2006]

[47] SOCHOR, Jiří. Údržba softwaru. *Zpravodaj ÚVT MU*. Brno: Masarykova universita, Fakulta informatiky. 1996, roč. VI, č. 3, s. 15-20, ISSN 1212-0901. Dostupný z: http://www.ics.muni.cz/zpravodaj/articles/61.html [www dokument]

[48] TAKKINEN, Juha <juhta@ida.liu.se> . *Famous Computer Scientists Related to the Laboratory for Intelligent Information Systems*. [Slavní počítačoví vědci spojení s laboratoří inteligentních informačních systémů] Linköping, Švédsko : Laboratory for Intelligent information Systems, Dept. of Computer and Information Science, Linköping universitet, 27. 6. 2001. 28 s. [cit. s. 15, 16]. Dostupný z: http://www.ida.liu.se/~juhta/publications/famous_cs_related_to_iislab.pdf [dokument ve formátu PDF]

[49] *Terminologický slovník*. Česká společnost pro systémovou integraci. [online]. Dostupný z: http://www.cssi.cz/all_terminologie.asp [cit. 20. 10. 2006]

[50] TICHÁ, Soňa. Dokumentace k Oracle 8i. Ostrava: FEI VŠB TU Ostrava, Katedra informatiky [www dokument]. Dostupný z: http://www.cs.vsb.cz/ticha/oracle/oraq1.htm [cit. 7. 11. 2006]

[51] VAVRUŠKA, Jindřich. ETL a kvalita dat. *IT systems*. 2003, Příloha 3/2003: Data warehousing a Business Inteligence. [www dokument]. Dostupný z: http://www.systemonline.cz/site/data-warehousing/kvalita4.htm [cit. 2. 11. 2006]

[52] VIERA, Robert. SQL *server 2000: programujeme profesionálně*. Praha: Computer Press, 2001, 1206 s., ISBN: 8072265067

[53] VÍT, Ondřej. *Data pod kontrolou: řešení datového skladu v prostředí Celní správy ČR*. Přednáška na konferenci ISSS/LORIS/V4DIS. Praha, 3. 4. 2006. Dostupný z: http://www.isss.cz/archiv/2006/download/prezentace/vit.ppt [dokument ve formátu ppt]

8 SEZNAM POUŽITÝCH OBRÁZKŮ A TABULEK

8. 1 Obrázky

8. 2 Tabulky