Analysis of Data Virtualization & Enterprise Data Standardization in Business Intelligence

Laljo John Pullokkaran

Working Paper CISL# 2013-10

May 2013

Composite Information Systems Laboratory (CISL)
Sloan School of Management, Room E62-422
Massachusetts Institute of Technology
Cambridge, MA 02142

Analysis of Data Virtualization & Enterprise Data Standardization in Business Intelligence

by

Laljo John Pullokkaran

B.Sc. Mathematics (1994)
University of Calicut

Masters in Computer Applications (1998)
Bangalore University

Submitted to the System Design and Management Program
in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Engineering and Management

at the

Massachusetts Institute of Technology

May 2013

Signature of Author____

Laljo John Pullokkaran
System Design and Management Program
May 2013

Certified by____

Stuart Madnick
John Norris Maguire Professor of Information Technologies, Sloan School of Management
and Professor of Engineering Systems, School of Engineering
Massachusetts Institute of Technology

Accepted by ____

Patrick Hale
Director
System Design & Management Program

Contents

Analysis of Data Virtualization

&

Enterprise Data Standardization in Business Intelligence

by

Laljo John Pullokkaran

Submitted to the System Design and Management Program
on May 20 2013, in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Engineering and Management

Abstract

Business Intelligence is an essential tool used by enterprises for strategic, tactical and operational decision making. Business Intelligence most often needs to correlate data from disparate data sources to derive insights. Unifying data from disparate data sources and providing a unifying view of data is generally known as data integration. Traditionally enterprises employed ETL and data warehouses for data integration. However in last few years a technology known as “Data Virtualization” has found some acceptance as an alternative data integration solution. “Data Virtualization” is a federated database termed as composite database by McLeod/Heimbigner's in 1985. Till few years back Data Virtualization weren’t considered as an alternative for ETL but was rather thought of as a technology for niche integration challenges.

In this paper we hypothesize that for many BI applications “data virtualization” is a better cost effective data integration strategy. We analyze the system architecture of “Data warehouse” and “Data Virtualization” solutions. We further employ System Dynamics Model to compare few key metrics like “Time to Market” and “Cost of “Data warehouse” and “Data Virtualization” solutions. We also look at the impact of “Enterprise Data Standardization” on data integration.

Thesis Advisor: Stuart Madnick
Title: John Norris Maguire Professor of Information Technologies, Sloan School of Management and Professor of Engineering Systems, School of Engineering
Massachusetts Institute of Technology

Acknowledgements

This thesis wouldn't have come around without the support of Professor Stuart Madnick, my thesis advisor. While I was moving around in the maze of big data, it was Professor Madnick who brought me back to the world of data integration challenges. Without the guidance of Professor Madnick, I wouldn't have gotten in to enterprise data standardization issues. I am grateful for Professor Stuart Madnick's patience and guidance.

I would like to thank Robert Eve and Bob Reary of Composite software for their guidance and inputs on adoption of data virtualization in the real world. Robert and Bob worked with me through multiple sessions on the value proposition of data virtualization and on how data virtualization complements and competes with ETL and data warehouse.

If it wasn't for Twinkle, my wife, I would have dropped out from SDM program. Working full time in a startup and working on SDM at the same time left me with very little time to take care of Anna, our 2 yr. old. I am really fortunate to have somebody who put up with two years of absentee parenting.

Chapter 1

Introduction

1.1 Why look at BI?

Success and Failure of most organizations can be traced down to the quality of their decision making. Data driven decision making has been touted as one of the ways to improve decision making. Data driven decision making has the potential to improve the quality of decisions as decisions would be based on hard facts as opposed to gut feel or hunch. Data driven decision making requires collection and analysis of data.

Every organization gathers data on various aspects of its business whether it is sales records or customer calls to its support department. Closer analysis of data and correlation of data from various departments can provide key facts about current state of business and historical trends. Data can be used for predictive and investigative analysis. Customer churn analysis to market basket analysis, data can provide key insights in to business. In the paper titled “Customer Churn Analysis in the Wireless Industry: A Data Mining Approach” Dr. Ravi S. Behara claims to have achieved 68% accuracy in predicting possible customer churn in wireless industry using fifty thousand customer records and Naïve Bayes algorithm.

From 1960 organizations have been employing Decision Support Systems, DSS, to aid decision making. Over the years DSS has evolved in to “Business Intelligence”. Business Intelligence, BI for short, is the set of process and technologies that derives meaningful insights out from raw data. Almost all business organizations employ BI for planning and forecasting. Last couple of decades has seen more applications of BI in tactical and operational decision making. These days for most business organizations, BI is not a choice but a must have investment to survive in competitive market place. According to IDC estimates the size of BI market was around 33.4 billion for the year of 2012.

BI has become a strategic necessity for most business organizations. However literature survey indicates that BI projects have longer implementation cycles and is rather inflexible to accommodate changes. Improving BI implementation cycle and flexibility could allow for more successful BI projects thus potentially accelerating data driven decision making in organizations.

1.2 Significance of data integration in BI

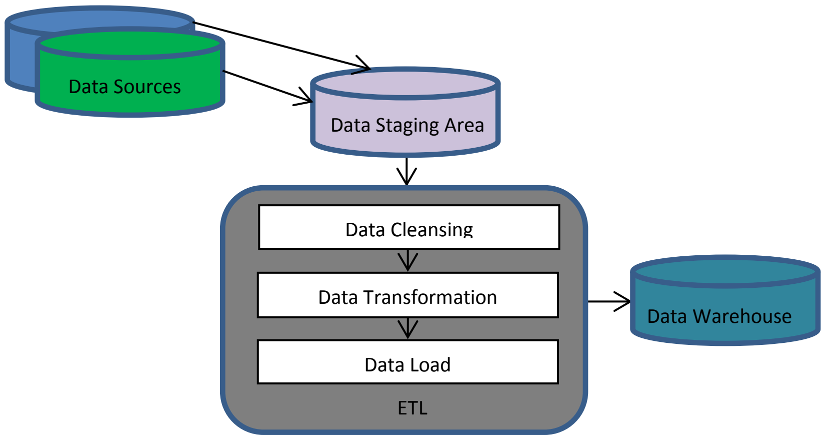
One of the key steps in BI process is the extraction and correlation of data from various data sources employed by an organization. In todays’ globalized market most organizations have multitude of information repositories. Human Resources, Sales, Customer Management and Marketing will all have information systems for their needs. Often each of these departments will have multiple databases and applications; these days with the adoption of SAAS, more and more data is kept in different cloud offerings along with some databases in premise. It is common these days to find small and medium size business to keep their Sales data in "SalesForce”, human resource in “WorkDay”, financials in “NetSuite” along with databases in premise. Extracting data from all these systems is a necessary step of data integration.

Data from production data stores can be extracted in many different ways, “pull” and “push” are the common methodologies employed. In “pull” model, data is moved out from production store when needed whereas in “push” model every time data changes in production data store, changed data is propagated out. Data in production data stores often contain errors introduced by data entry. Also different departments may have different definitions for same business entity, like customer, these semantic differences in what data describes needs to be reconciled before data can be correlated. Data cleansing deals with errors in production data and prepares it for data transformation. Cleansed data may be transformed to normalize various attributes so that correlation can be performed. Cleansed and transformed data from various departmental data stores is correlated to derive big picture view of business. This correlated data may sometimes be further transformed by filtering or by adding additional details to it or by aggregating information. Analysis and reports are then generated out from the above data set. This process of extracting data from production data stores, cleansing them, transforming them, correlating them is generally known as data integration.

1.3 Data Warehouse vs. Data Virtualization

Data Integration is an essential step for BI applications. Traditional BI approaches physically moves data from origin data sources to specialized target data stores after going through data cleansing, transformation and de normalization. This process of moving data physically from origin data source is known as Extract-Transform-Load (ETL). Another variant of this process known as Extract-Load-Transform is also used sometimes for data movement. Fundamental difference between ETL and ELT is the order in which transformation is applied to the data. In ELT, data is extracted then loaded in to data warehouse and then transformation is applied on the data whereas in ETL data is extracted, transformed and then loaded in to data warehouse.

The specialized target data store which is used to store integrated data is termed as Data Warehouse. Data warehouse is a central database which stores integrated data so that reports and analysis can be run on the data. Data warehouse databases often support pre aggregation of data and is optimized to perform queries on large data set. The data set in the data warehouses are often known to host terabytes of data running in to billions of records. Many data warehouse solutions employ custom hardware and massively parallel computational architecture to speed up query execution.



ETL & Data Warehouse
Figure: 1

Data Virtualization is a relatively new approach to data integration. Data Virtualization does not move the data physically as part of data integration instead all of the data cleansing, data transformation and data correlation is defined in a logical layer which is then applied to data as they are fetched from origin data source while generating reports. Data Virtualization seems to have shorter implementation cycle and better agility thus enabling more analysis and hypothesis testing.

1.4 BI & Enterprise Data Standardization

Enterprises have been trying to reduce the cost of data cleansing by trying to standardize the data sources and data definition, this is known as "Data Consolidation" or "Enterprise Data Standardization". "Enterprise Data Standardization" has at least two facets to it. Consolidating on a single or few vendors for Data Sources and standardizing the semantic interpretation of common attributes. "Enterprise Data Standardization" has long term benefits but requires discipline in order to maintain single semantic definition of various business entities and may often compromise agility.

1.5 Research Questions

In this thesis we examine the merits of "Data Virtualization" and "Enterprise Data Standardization" for BI applications. We contrast these approaches with traditional Data warehouse approach. Thesis examines differences in system architecture, and system dynamics to see how it impacts business drivers like "Time To Market" and "Cost".

Hypothesis 1

Data Virtualization is a suitable data integration technology for BI applications that doesn't require analysis of large amounts of data, and can result in significant "Cost" and "Time To Market" savings.

Hypothesis 2

"Enterprise Data Standardization" significantly reduces data cleansing cost and time.

Chapter 2

Overview of Business Intelligence

Organizations employ BI for aiding their decision making process. Organizational decision making can be broadly classified in to strategic, tactical and operational decisions. Organizations employ BI in support of all three types of decision making.

2.1 Strategic Decision Support

Strategic decision support has mainly to do with decisions that affect long term survival and growth of the company. Decision making at this level could be decisions like “Should we enter this new market?” “Should we grow organically or through acquisition”, “Should we acquire this company”. BI projects in this space would most often require access to historical data and ability to test hypothesis and models.

2.2 Tactical Decision Support

Tactical decision making is normally focused on short and medium term decisions. These decisions are made usually at individual organizational level. This would include things like demand forecasting, analyzing customer churn data etc. Tactical decision support often employs data mining and predictive analytics.

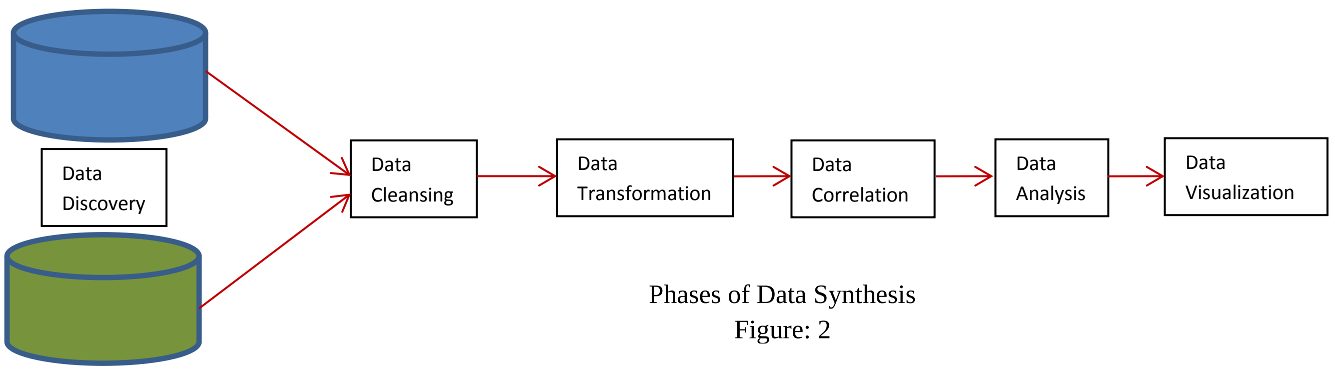
Data Mining is the process employed to discover hidden patterns in data. For example data mining can be employed to identify reasons and trend lines behind customer churn. Predictive analytics on the other hand is employed to anticipate future outcomes. Which products can I sell to existing customers, what kind of offer will pacify an angry customer... are some examples of predictive analytics. Historical data will be mined to extract patterns which is then modeled and deployed; attributes of various business entities are then continuously monitored using these models to identify trend lines.

2.3 Operational Decision Support

Operational decision support deals with daily or hourly operational decision making. Some examples of such decisions are how many materials to send to distributor tomorrow, production quota for today, how many materials to move from inventory to production line. Operational Decision Making requires real time data collection and integration of that real time data with historical data and models generated by tactical decision support systems.

2.4 Business Intelligence and Data Integration

One common theme among strategic/tactical/operational decision making is the ability to correlate data from different sources. For operational efficiency it is natural for each department to have its own data store; sales organization is primarily concerned about their sales work flow whereas support organization is primarily concerned about support tickets work flow. However to understand hidden opportunities and threats like cross selling products to existing customers an analyst needs to co relate data from different departmental systems. Following describes the major logical steps in the evolution of raw data to useful business insights.



Phases of Data Synthesis
Figure: 2

2.4.1 Data Discovery

One of the first problems to tackle for BI and data integration projects is taking inventory of all the available data across the organization that is relevant to the BI project. Due to decentralized information system organizations and due to mergers and acquisitions, business often accumulates large number of data repositories. In the book titled "Data Virtualization", September 2011, Judith R. Davis and Robert Eve describe a fortune 500 company that had hundreds of application database tables spanned across six different regional information systems. In mid-size and large companies it is common to have large number of application database tables. Understanding what each of these tables is about and how they relate to other tables in the organization is the first task in data integration.

2.4.2 Data Cleansing

The quality of the analysis depends on the quality of the source data. In practice it is observed that source data often contain errors introduced during data entry. In addition with heterogeneous systems it is often the case that the semantic definition of business entity could differ across systems. Data Cleansing is the process of removing errors from source data.

Erroneous data

During data entry in to the production system, errors may creep in. These errors need to be recognized and corrected before data can be integrated. According to the paper titled "Data Cleaning: Problems and Current Approaches" by Erhard Rahm, Hong Hai Do, University of Leipzig, Germany, erroneous data in data sources can be broadly classified as follows:

- Illegal Values
  These are values that violate the boundary conditions for values.
  Example: Order Date: 15/01/2012
  The date format is MM/DD/YYYY.

- Dependency Violation
  If one attribute of data is true then it would violate some other attribute.
  Example: age 10, birth date 01/01/1975

- Uniqueness Violation
  For attributes that are supposed to be unique, multiple records have same value for unique attribute.
  Example: Emp No: 55, Emp Name: Joe Smith
  Emp No: 55, Emp Name: Ron Smith

- Referential Integrity Violation
  Referential Integrity violation can happen when foreign key references do not exist.
  Example: Emp No: 55, Emp Name: Joe Smith, Emp Dept No: 27
  However dept no 27 doesn't exist.

- Missing Values
  Some of the attributes may not have values filled in.
  For example:
  Customer Acquisition Date: NULL
  Customer Name: Joe Bode

- Miss spellings
  During data entry, spelling mistakes may creep in.
  For example in Sales Lead Table, inside sales contact's email id is miss spelled.

- Cryptic Values or Abbreviations
  Example: Occupation: SE (instead of Software Engineer)

- Embedded Values
  Example: Name: "Joe Smith San Francisco"

- Misfielded Values
  Example: City: USA

- Duplicate Records
  Example: Emp No: 55 Emp Name: J Smith, Emp No: 55 Emp Name: J Smith

- Contradicting Records
  Example: Emp No: 55 Emp Name: Joe Smith Dept Id: 10, Emp No: 55 Emp Name: Joe Smith Dept Id: 5

Semantic Errors

When data is integrated from multiple sources errors arise often due to differences in the definition of business entities. While correlating data from different departmental information systems two problems arises:

1. Recognizing that the two different tables are describing same business entity.
2. Reconciling the difference in attributes captured on those business entities.

Reconciling these differences and compensating as needed would often require data transformation.

For example a CRM system may capture different aspects of “customer” when compared to a “sales” system.

Sales Lead Table:

Prospect ID	Name	Address	Sales Contact	Opportunity	Sales Stage	Existing Customer
21	Barn & Smith	500 Mass Ave, MA	Joe Bode	500000	POC	TRUE

Customer Table in CRM:

ID	Name	Address	Contact	Revenue	Support Contract	Support Tickets Open
21	Barn & Smith	500 Mass Ave, MA	Tony Young	1000000	12/31/2014	12002

Both Sales Lead Table and Customer Table are describing different attributes of existing customer. “Sales Lead” table is describing new sales opportunity where as “Customer Table” in CRM is describing attributes of existing customer that is relevant for providing support. Semantic differences arise in this specific example because each department is focused on its immediate function. As part of data cleansing these two tables could be broken up in to three different tables in which one table would focus exclusively on aspects of customer that doesn’t mutate across departments and other table that describes aspects of customer that is different to each department.

Customer Master Table:

ID	Name	Address	Existing Customer	Products Sold	Revenue
21	Barn & Smith	500 Mass Ave, MA	TRUE	Data Integrator	100000

Sales Lead Table:

Customer ID	Sales Contact	Opportunity	Sales Stage
21	Joe Bode	500000	POC

Customer Table in CRM:

Customer ID	Support Contact	Support Contract	Support Tickets Open
21	Tony Young	12/31/2014	12002

Phases of Data Cleansing

Data cleansing is a multi-phased procedure which typically involves:

- Error Detection
  Error detection requires automated and manual inspection of data. Often metadata would help in this error detection process. One of the major sources of problem is recognizing that two different representations of data are actually describing the same business entity. Data profiling tools can help in this process.

- Error Correction
  Once data errors are identified and classified then they need to be corrected. Correcting erroneous data entry could be tedious and manual process. Correcting schema differences requires more analysis and requires knowledge of use cases of the consumers of the data. Once correct schema representation is identified then transforming data into the required schema could be automated.

- Data Verification
  The correctness and effectiveness of data cleansing needs to be measured and based on the result may often require redoing Error Detection and Error Correction steps. Data verification could employ tools to check if all data items follow boundary constraints. Semantics error correction often requires manual verification.

- Replacing Source Data
  In some cases, like in cases where data is wrong due to data entry problems, data in the source systems needs to be replaced with the corrected data. This may impact the applications that rely on the operational data store.

2.4.3 Data Transformation

After data is corrected of semantic and data entry errors, data may need to be transformed further so that they can be correlated correctly. Data transformation is required primarily due to the following:

- Data type incompatibilities
  Usually data from different data sources are filtered, and joined together as part of data integration. Equivalence of type of data becomes important when two pieces of data is compared. If data type definitions are different then it would result in data loss. Data type incompatibility primarily arises from three sources:
  1. Usage of different data types for same piece of data
     Different departmental systems may choose different data types for same piece of data. For example customer SSN, some department may choose to store it in a String data type whereas others may store it in an integer data type.

  2. Not all data types are supported by all data sources
     A common problem found in relational data bases is that not all data types defined by SQL are supported by all vendors. For example OLAP databases typically won't support LOB data types. Similarly some databases support Boolean data types whereas others don't.

  3. Same data type in different data source may have differing definitions
     Approximate numeric/floating point types have the common problem that their precision and scale may not be same across different data sources. Similarly for variable length data types like char in SQL, different departments may have chosen different lengths.

- Data Formatting differences
  Data formatting needs to be normalized before integration. Dates and Time Stamps are common sources of such problems. Different departments may have different formats for these. For example in US most organizations follow MM-DD-YYYY format whereas in UK and its former colonies DD-MM-YYYY format is followed.

- Unit differences
  The units of data may be different across different departments. Data for money is one such common source of error. For example, departmental information systems in Europe may express revenue in Euro whereas in US revenue may be expressed in dollars. Before joining these two data items they need to be normalized to a common currency.

2.4.4 Data Correlation

Once data has been cleansed and normalized, data from different systems can be correlated. Typically correlation of data would involve some or all of below:

- Filtering
  Select only a sub set of available data based on some condition. For example, sales data for last quarter and product inventory for last quarter.

- Joining
  Data from different tables that satisfies some criteria may need to be joined together.
  For example to find out sales data for each product for last quarter, data from sales needs to be joined with product table.

- Aggregation
  Data may need to be partitioned based on some attributes and then minimum/maximum/total/average of each partition may need to be calculated. For example, for last quarter, group sales data by products and then find total sales by region.

2.4.5 Data Analysis

Once data has been correlated it is ready for analysis. Business analysts can run hypothesis testing or pattern discovery on the data. For example, did price hike contribute to increase in customer churn? Data analysis can be exploratory or confirmatory; in exploratory approach there is no clear hypothesis for data analysis whereas in confirmatory approach hypothesis about data is tested.

2.4.6 Data Visualization

Results of the analyses need to be captured visually so that findings can be easily communicated. Pie charts, graphs etc. are used for data visualization. Specialized software like Tableau, clickview allow for new ways of data visualization. Common focus of visualization is on information presentation.

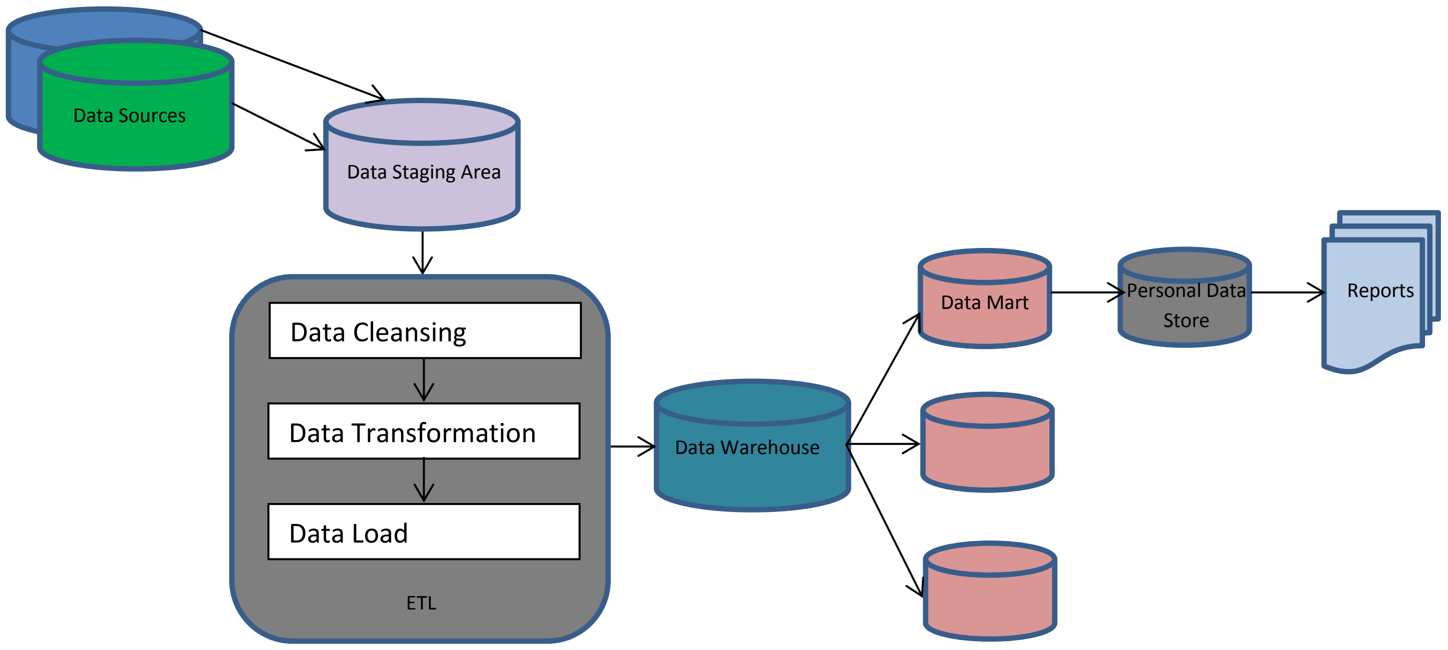
Chapter 3

Traditional BI Approach – ETL & Data warehouse

Most of the current implementations of BI systems involve ETL and Data warehouse. In this chapter we provide an overview of how ETL & Data warehouses are used for BI.

Following describes the key elements of traditional BI systems:

- Data Staging Area
- ETL
- Data Warehouses
- Data Mart
- Personal Data Store (PDS)



Traditional BI
Figure: 3

3.1 Data Staging Area

Data staging area basically captures data from the production systems without altering data. Data is then copied from staging area to data warehouse after going through cleansing and transformation as part of ETL processing. Data copy rate from staging area to data warehouse most often won't match data change rate in the production data sources. Data staging area would need to keep data around till it is copied in to data warehouse. Some of the key issues regarding data staging area are:

- Should staging area reflect source system's tables or should it reflect the schema of planned data warehouse?
- How often to move data from production systems to staging area?

3.2 ETL

ETL process theoretically involves moving data from production systems, transforming it before loading in to target data store like data warehouse. But in many deployments data is first moved in to data staging areas and data is cleansed before it is transformed and loaded in to target data stores like data warehouse. Data movement itself could be either a PULL model or a PUSH model. In PULL model ETL tool, based on schedule defined by user, would copy data from production data systems or data staging area in to Data warehouse. In PUSH model data changes in production systems or data staging area would be propagated out and ETL would kick in for the data changes. PUSH model would typically employ 'Change Data Capture" technology. "Change Data Capture" employs technology to listen for changes in production systems and to propagate changes to other systems; for databases, "Change Data Capture" would look out for changes in "Redo Log" files and would propagate the changes to other systems that are interested in the data changes. According to a study by TDWI, The Data Ware Housing Institute, around 57% of all ETL is schedule based pull approach, 16% of ETL is push based and rest 27% of ETL is on demand.

Data transformation typically would involve applying business rules, aggregating or disaggregating the data. One important concern regarding ETL is the performance of ETL. Most ETL tools would apply some sort of parallel data fetch & transportation mechanism to improve the speed.

3.3 Data Warehouse/DW

Data warehouse is a central repository that stores cleansed and correlated data. Data warehouse is normally a central database which supports SQL. Many of these databases employ specialized hardware to accelerate query performance. Data warehouse usually stores data for long periods of time and changes in source data gets appended to already existing data thus providing a chronological order of changes. Data warehouse provides some key benefits to organization:

- Off loads data analysis workloads from data sources
- Provides an integrated view of data
- Provides a consistent view of data
- Provides historical perspective on data

Data warehouses ends up storing large amounts of data, often billions of records running in to terabytes. This large size of the data introduces some unique challenges for executing queries and for applying changes to schemas. Organization of the data has a key role to play in query performance and schema flexibility. Over the years various different architectures has evolved for data organization in data warehouses. Some of the important architectural considerations are:

- Normalized vs. Denormalized Schema
- Star Schema
- Snowflake Schema

3.3.1 Normalized Schema vs. Denormalized Schema

Normalization is the process of grouping attributes of data which provides a stable structure to data organization. Normalization reduces data duplication and thus avoids data consistency issues. Relational Database theory has a number of different popular normalization forms. Most of the production data stores, Online Transactional Processing System, would employ normalized schemas. However for BI applications, normalized schema introduces a challenge. Normalization would group different attributes in to different tables; since BI needs to correlate data from different tables, BI applications would have to join data from different tables. Since data warehouses have large data set, these join operations tend to be expensive.

An alternative strategy is to employ normalized schema for production systems and use de-normalized schema for data warehouse. In this model BI apps would avoid costly joins since data is de-normalized. There are number of de-normalization techniques employed by industry; the popular de-normalization techniques include:

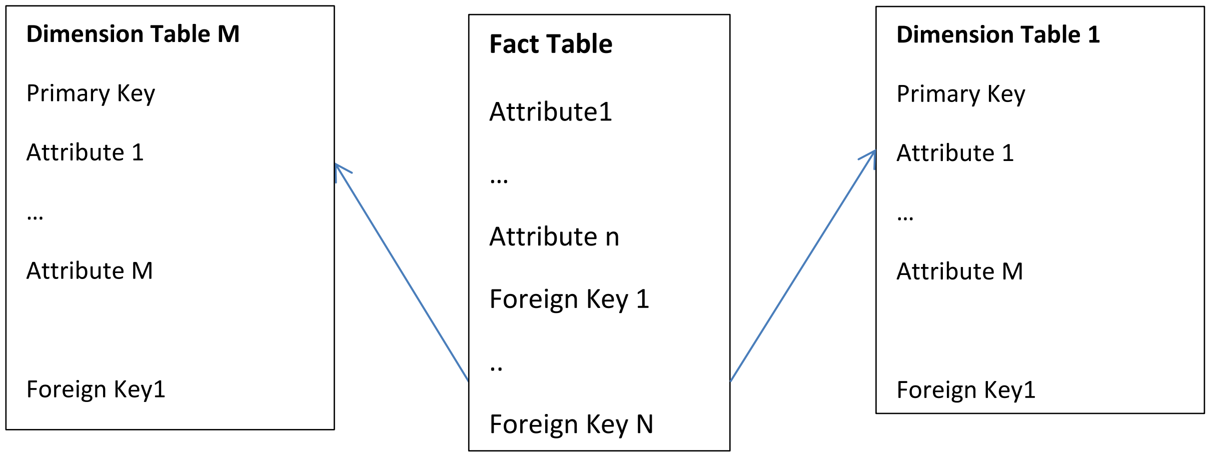
- Materialized Views
  Materialized View refers to the caching of data corresponding to a SQL query. The cached data is treated as a regular database table and all operations that are supported by regular table are supported on materialized view. Indexes can be created on the materialized views just like in regular table. Materialized Views reduces the load on data warehouse DB as frequently used queries can be cached as materialized views thus enabling reusability and reducing work load.

- Star & Snow Flake Schemas
  Star & Snow Flake schemas follow dimensional data model which groups data in to fact table and dimension tables. Fact table would contain all critical aspects of the business and dimension table would contain non critical aspects of the business. Fact table would be in general large data sets and dimension table would be small data set.

- Prebuilt Summarization
  Data warehouses supports multi-dimensional data structures like cubes, which allows for aggregating data across different dimensions. Once cubes are created then multiple analysis could share cubes to analyze data across different dimensions.

3.3.2 Star Schema

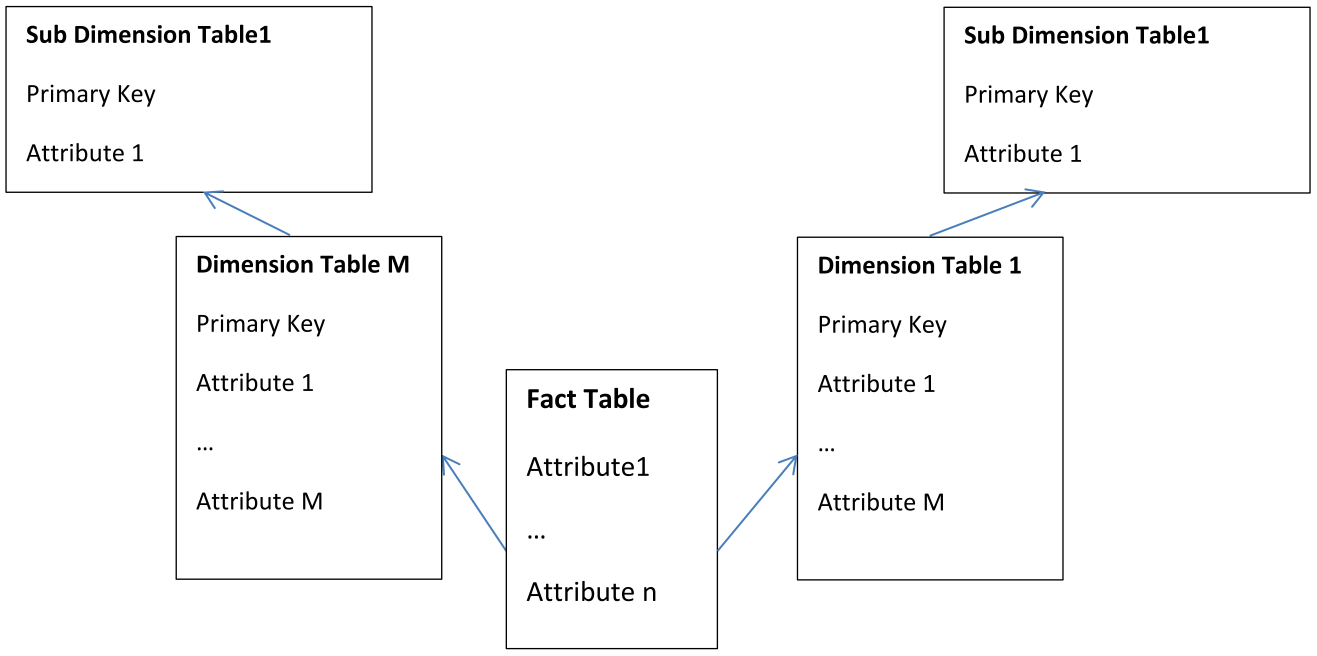
In Star Schema, data set is divided in to facts and its descriptive attributes. Facts are stored in a central fact table and descriptive attributes in a number of separate tables known as dimension tables. Central fact table would have foreign key references to dimension tables.



Data Warehouse – Example of Star Schema
Figure: 4

3.3.3 Snowflake Schema

Snowflake schema has a central fact table and multiple dimension tables. Each of the dimension tables may have sub dimensions. The sub dimensions are the key difference between Star and Snowflake schema. Compared to Snow Flake schema, star schema is easier to implement.



Data Warehouse – Example of Snow Flake Schema
Figure: 5

3.4 Data Mart

Data Mart is in general can be considered as a subset of data warehouse. Usually these are created per department. While data warehouse has a global scope and assimilates all of the data from all information systems, data mart is focused on a much limited scope, scope either contained by organization or by time.

Data Mart scope can be generally classified in to the following:

- Geography
  The focus here is on obtaining data that only relates to specific geographic areas like for example looking at data from Asia Pacific.

- Organization
  Focus is on organization like all data for sales dept.

- Function
  Data is bounded by function for example all data relates to customer interactions.

- Competitor
  Competitor data mart is focused on consolidation of all data relating to competitors.

- Task
  Consolidation of data needed for specific task like budget planning and forecasting.

- Specific Business Case
  Data needed for specific business case like customer churn analysis.

3.5 Personal Data Store/PDS

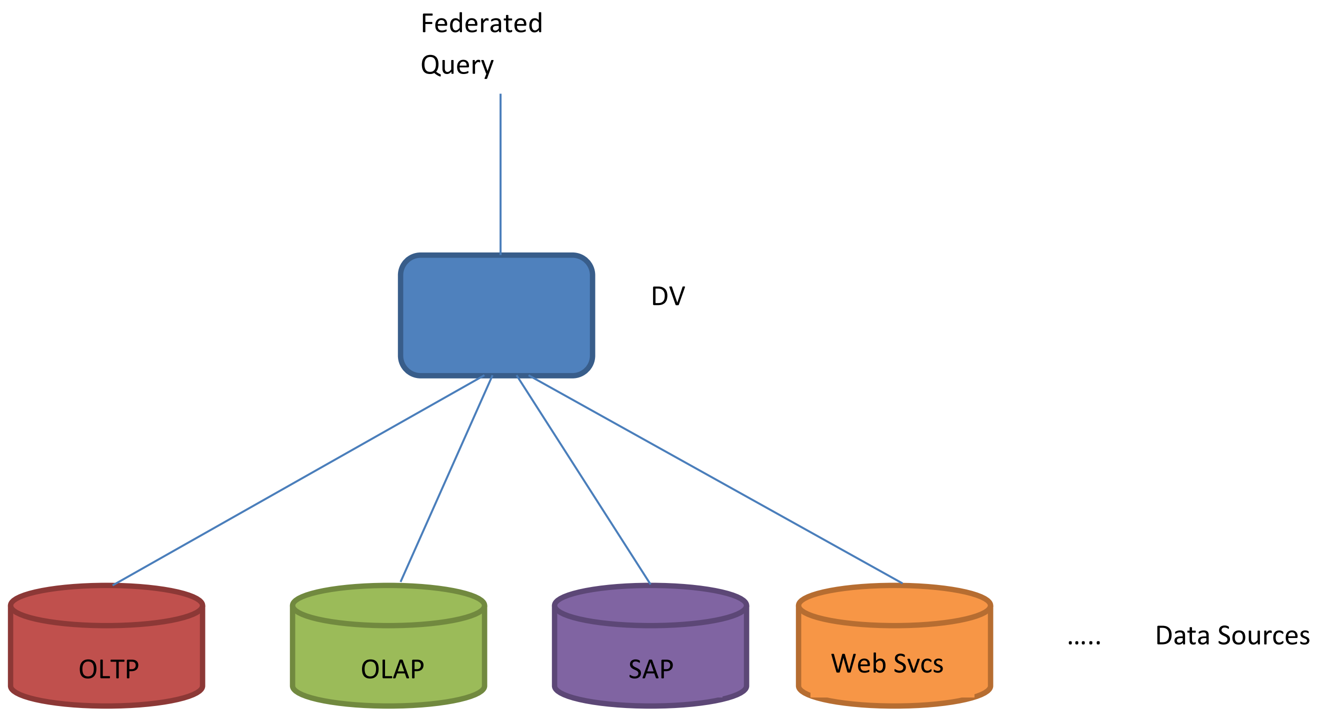
Personal Data Store is used mostly for personal uses or for a group of users, mostly by business analyst. Traditionally these data store tended to be spread sheet or flat file. These days many data bases and data warehouses provide personal sandbox to run experiments. The actual hypothesis testing is mostly done by analysts using personal data store.

Chapter 4

Alternative BI Approach - Data Virtualization

Data virtualization, DV for short, attempts to perform data cleansing, data transformation and data correlation as data moves out from production systems thus avoiding any intermediate storage. This is opposed to Data warehouse approach which physically changes data in each stage and loads it in to some data store. Typical Data virtualization platform requires:

- Ability to Discover data stored in data sources
- Ability to retrieve data from different data sources
- Ability to define views or virtual tables
- Ability to optimize federated query
- Ability cache data
- Fine grained security



Data Virtualization – Over View
Figure: 6

The above diagram depicts usage of DV platform for integrating data from databases, SAP and web services. Each phase of data analysis gets defined using virtual tables; please refer to section below on more details on virtual tables. Each phase uses data from previous phase by referring to virtual table from previous section. When analysis needs to be done DV compiles all of the definitions of virtual tables in to a single SQL, which is then compiled, optimized and executed.

4.1 Data Source Connectors

DV platform needs to bring data from disparate data sources. Different data sources have different access mechanisms. Data coming from data bases would need queries submitted through SQL using JDBC/ODBC connections, data coming from supply chain and financial apps would require their proprietary access mechanisms, fetching data from web services would require web service access methods. DV platform needs to have data access mechanisms for all these different data sources.

4.2 Data Discovery

Data may need to be retrieved from variety of data sources like databases, applications, flat files, web services. DV platform needs to understand the schema of data storage and relationship among them. For example with data source that uses data base, a DV platform needs to figure out all of different tables in the database, constraints defined on them, primary-key, foreign key relationships. Databases typically store this information in their system catalogs which could be used by DV. The metadata about these schemas needs to be stored with in DV platform so that when user submits a query, DV platform can properly execute the query by fetching data appropriately.

4.3 Virtual Tables/Views

Virtual Tables or Views are the result set of a stored query which can be used just like a regular database table. The table schema and table contents are defined by SQL. The table is considered virtual because table contents are not physically stored. Data for the virtual table is brought in from underlying database tables when query defining virtual table is executed.

For example "Inventory_Data" is a virtual table that has columns "productid", "inventory" and "cost". The contents of the table "Inventory_Data" comes from database table "Inventory_Transactions".

View Inventory_Data:

Select productid, inventory, currencyConvertToDollars(cost) as cost from Inventory_Transactions

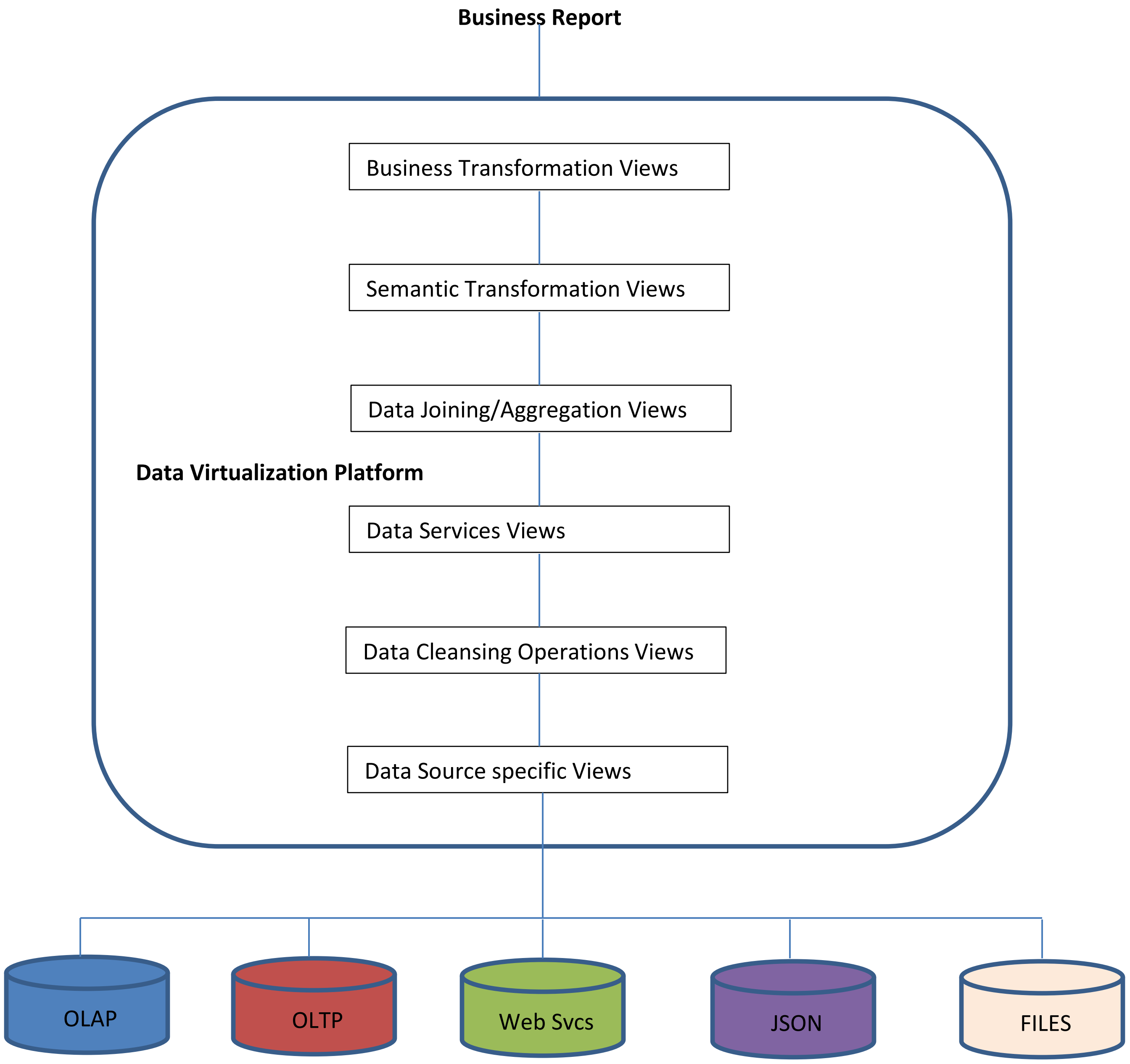
Inventory_Transactions Table in DataBase:

ProductID	Inventory	Cost	BackLog	Supplier
10	100	5000		XYZ Tech

In order to be used for data integration DV platform must do data cleansing, data transformation and data correlation. DW does each of these stages separately and produces physically transformed data at each stage. Unlike DW and ETL, DV platform would do all of these stages mostly in one step before producing the final report. DV platform needs to define data cleansing, data transformation and data correlation logic programmatically using SQL like query language. Typically this is achieved by defining views or virtual tables; typically users would define multiple such views at different levels of abstraction. When a report is generated data moves from data sources through the layers of these views cleansing, transforming, joining data before producing the report.

In some cases DV may use implicit virtual tables to expose data from data sources. This is because underlying data source may not be relational and instead could be hierarchical, multi-dimensional, key value store, and object data model; DV needs to convert all of these data models back in to relational data model and provide SQL as the language for data access. In such cases DV would expose underlying data as tables; for such implicit virtual tables, there wouldn't be any SQL defining the schema instead it would be natural mapping of underlying data in to relational able.

For example assume sales data is kept in "SalesForce.com". "SalesForce.com" uses Object data model and data is retrieved using SOAP API. Objects retrieved from "SalesForce.com" needs to be mapped to table. DV talks to "SalesForce.com" and gets the definition of all classes hosted in "SalesForce.com". Each of these classes is then exposed as a table by same name with class members becoming columns of the table.


Business Report
Business Transformation Views
Semantic Transformation Views
Data Joining/Aggregation Views
Data Virtualization Platform
Data Services Views
Data Cleansing Operations Views
Data Source specific Views
OLAP
OLTP
Web Svcs
JSON
FILES

Data Virtualization – Example of Virtual Tables
Figure: 7

The above diagram depicts deployment architecture for data integration using DV. In this example there are six different types of Virtual Tables one using the other as the source table. Data of interest from each data sources is first defined by data source specific views. Data cleansing views takes data from data source specific views and cleans them of data errors. Typically cleansing of data entry errors is done by passing data through custom build procedures. Transformation for semantic differences may be defined as separate virtual tables. Cleansed data is exposed as data services for each business entity. Data from the various business entities is correlated which is then transformed as required by analysis use cases. This data may optionally further go through additional transforms.

At each stage data from previous stage is accessed by specifying the virtual table from previous stage as the source of data. When a query is submitted to the top level virtual table (Business Transformation Views), DV would assemble the complete SQL by chaining user submitted query with the SQL that defines virtual tables from all stages.

4.4 Cost Based Query Optimizer

Since data cleansing, data joining, data transformation is all defined in virtual tables and since data is fetched from many different data sources, the relevance of optimal query execution becomes very important. Figuring out the cost of bringing data from different data sources and then planning query execution would reduce the query latency. Cost based optimizer is an important feature in reducing query latency time.

In the traditional data integration approach since the whole data is already in DW, the cost of fetching data from different data sources is avoided. Also DW often would have indexes built on data. Since DV doesn't store data, DV cannot take advantage of indexes. Most DW databases would have cost based optimizers, but focus of DW optimizer is to cut disk access time whereas in DV the focus of optimizer is to push work load as much as possible to backend production systems.

4.5 Data Caching

Ideally all of the data cleansing, joining and transformation should happen as data flows out from production systems; however this approach has some side effects:

- Query Latency may increase
- May impose load on production data stores
- May loose historical data

To avoid these problems sometimes it may be desirable to load intermediately transformed data in to data stores; this is effectively data caching. For example the output of data cleansing may be cached to avoid the load on production systems. Caching data needs to take in to account the rate of change of data in data sources.

4.6 Fine grained Security

Since DV platform needs to bring data from different data sources, it needs to manage the authentication and authorization to access various data sources. Similarly DV platform needs to return data to users based on their credentials. This would often require data masking and data filtering based on user authorization.

ETL process also needs to access data from production systems. Based on whether pull or push model is employed ETL needs to manage authentication and authorization to back end production systems. Similarly DW needs to deliver data based on authorization. In terms of security requirements there is actually not much difference between DV and ETL-DW.

Chapter 5

BI & Enterprise Data Standardization

Quality of results of data analysis depends to large extend on quality of the data. Data cleansing is the process to make sure that data is of good quality. Data cleansing is costly and often times requires manual effort. Enterprise data standardization is the process employed to reduce data errors and to improve data quality. While bringing together data from disparate systems there are three sources of data errors:

- Data type incompatibilities
- Semantics incompatibilities in business entity definitions
- Data entry errors

5.1 Data Type Incompatibilities

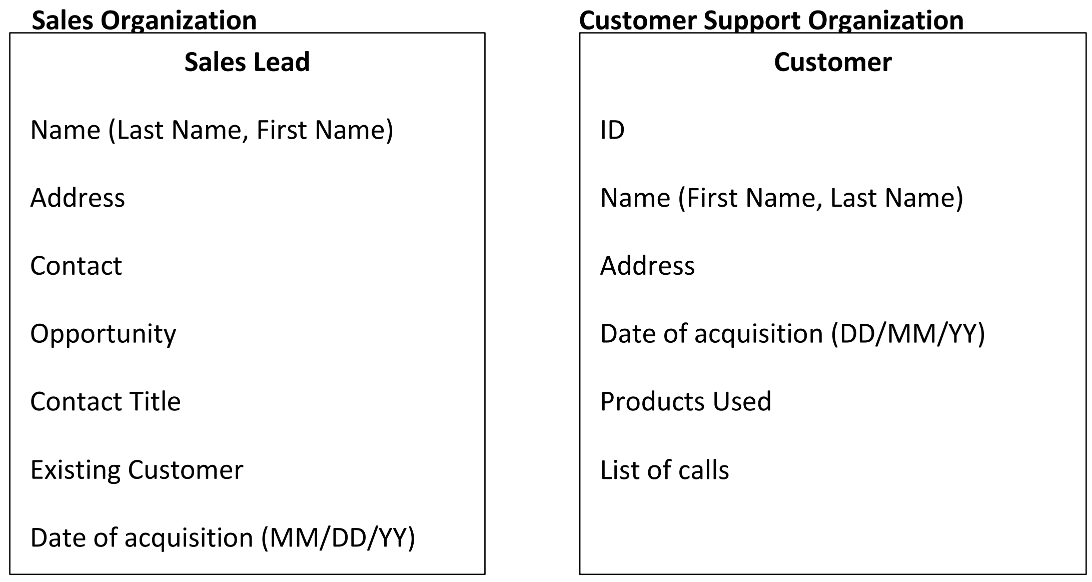
Data type mismatches can occur while correlating data from data sources of same data model; the problem gets magnified while correlating data from data sources of differing data models. Data sources belonging to same data model, like relational databases, is shown to have type incompatibilities as often there are some class of data types whose definition is left to implementation. For example for approximate numeric types in SQL, the precision and scale supported by different relational data bases are different; also the behavior of integral promotion/demotion (Ceil vs. Floor) for approximate numbers is implementation specific. Similarly not all relational data sources support Boolean data types. Mismatches in support for date and timestamps are also seen with relational data sources.

Collation Sequence is another source for issues while bringing data from different data sources. For example SQL standard leaves collation sequences support to specific implementations. A typical cause of problem is how case sensitivity and trailing spaces are handled across different relational systems. For example is ‘a’ considered equal to ‘A’ and ‘a’ considered equal to ‘a ‘?

These problems of type definition mismatches and collation sequence mismatches gets compounded when data sources belong to different data models, like joining data from relational data base with that of object data model.

5.2 Semantic Incompatibilities

Semantic incompatibilities refer to differences in the definition of various business entities which leads to data incompatibility. Typically semantic incompatibilities arise because different business units have different focus on a given business entity and hence each department include only those aspects of entity that is most relevant to them. For example, the definition of customer may vary from Sales organization to Customer Support Organization.


Sales Organization
Sales Lead
Name (Last Name, First Name)
Address
Contact
Opportunity
Contact Title
Existing Customer
Date of acquisition (MM/DD/YY)
Customer Support Organization
Customer
ID
Name (First Name, Last Name)
Address
Date of acquisition (DD/MM/YY)
Products Used
List of calls

Enterprise Data Standardization – Semantic Incompatibilities
Figure: 8

While bringing data from different data sources, semantic equivalence of different entities needs to be recognized. For example data integrator need to recognize that "Sales Lead" table in a relational database of "Sales" organization is referring to potentially existing customers which is also defined in "Customer" table in "Customer Support Organization". Recognizing that there is a semantic incompatibility is the first step. The second step is to decide how to bridge the semantic gap.

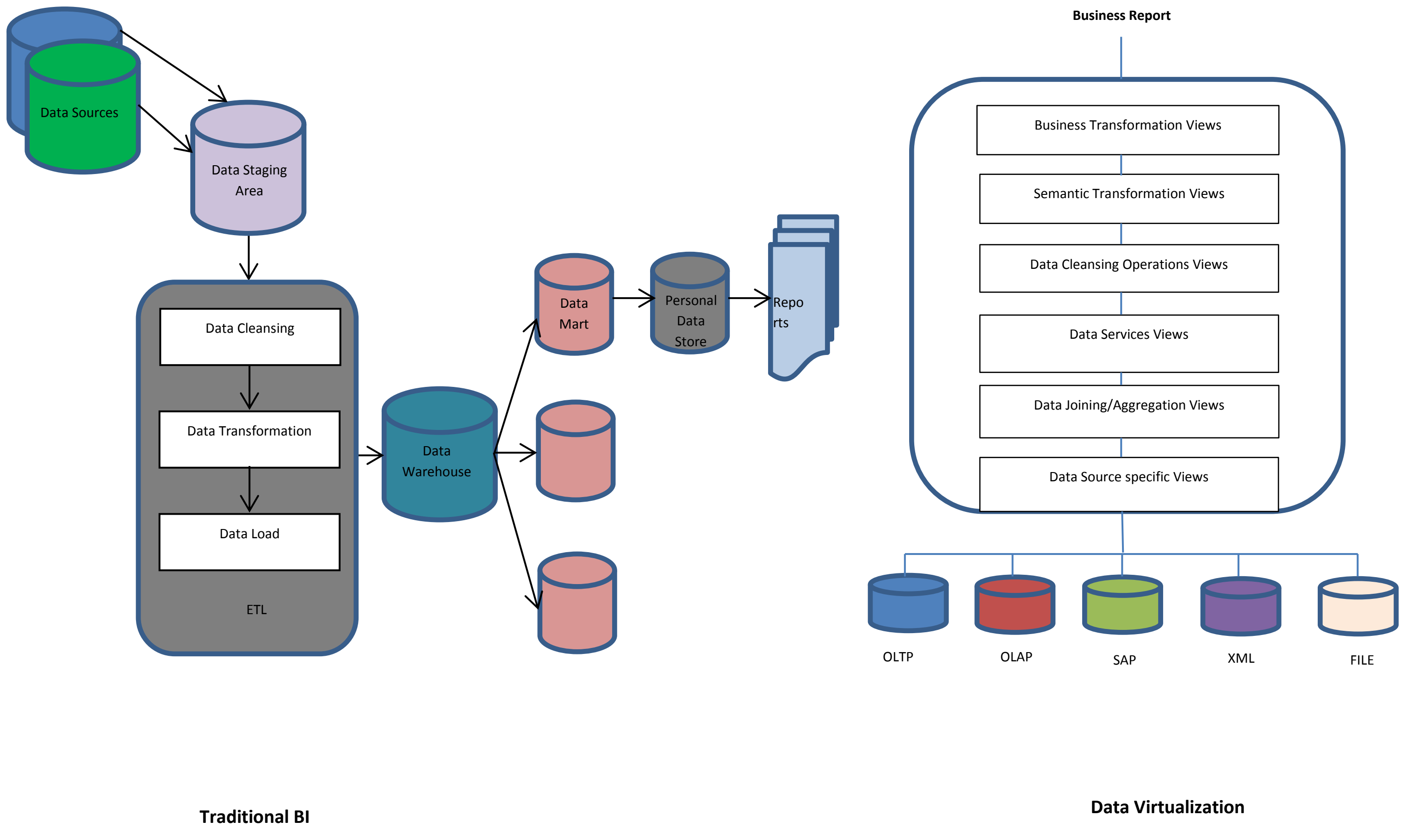
5.3 Data Standardization/Consolidation

Data type mismatches can be reduced if technology vendor for data sources is limited to few. There are many technology vendors who provide OLTP and OLAP data bases. Consolidating on a single vendor for OLTP and OLAP may not prevent all of data type issues since many data sources are neither OLTP and or OLAP. Reconciling semantic incompatibilities require companywide process initiative and discipline. Some organizations have adopted Master Data Management as a way to provide single definition of business entities.

Chapter 6

System Architecture Comparison

In this section we compare system architectures of traditional data integration approach and Data Virtualization. System complexity is one of the main factors that affect project cost and schedule. We use form centric view of system architecture and Dependency Structure Matrix (DSM) to compare relative complexities of these system architecture. We further use DSM to calculate impact of “Enterprise Data Standardization” on system’s complexity.

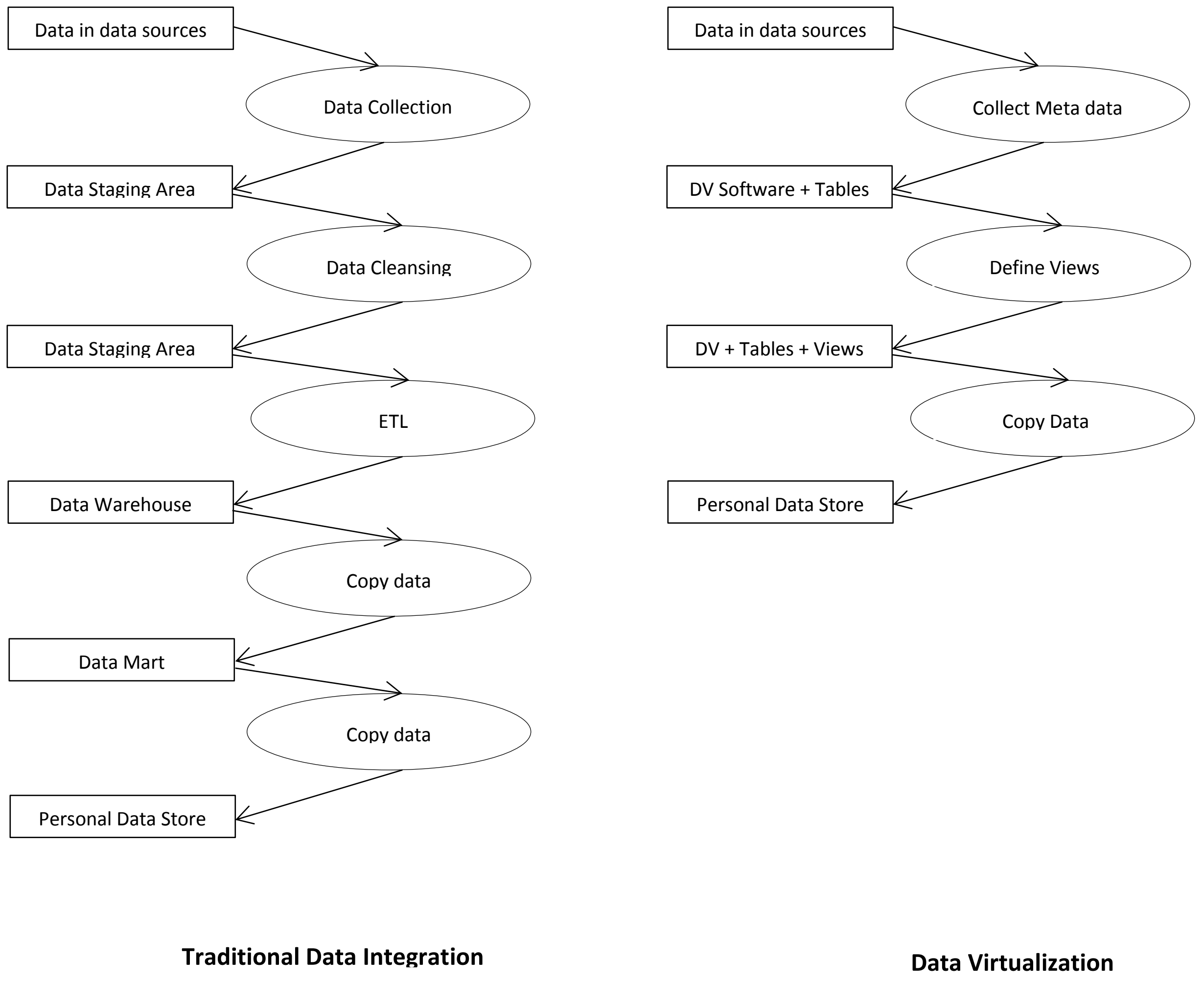


System Architecture Comparison

Figure: 9

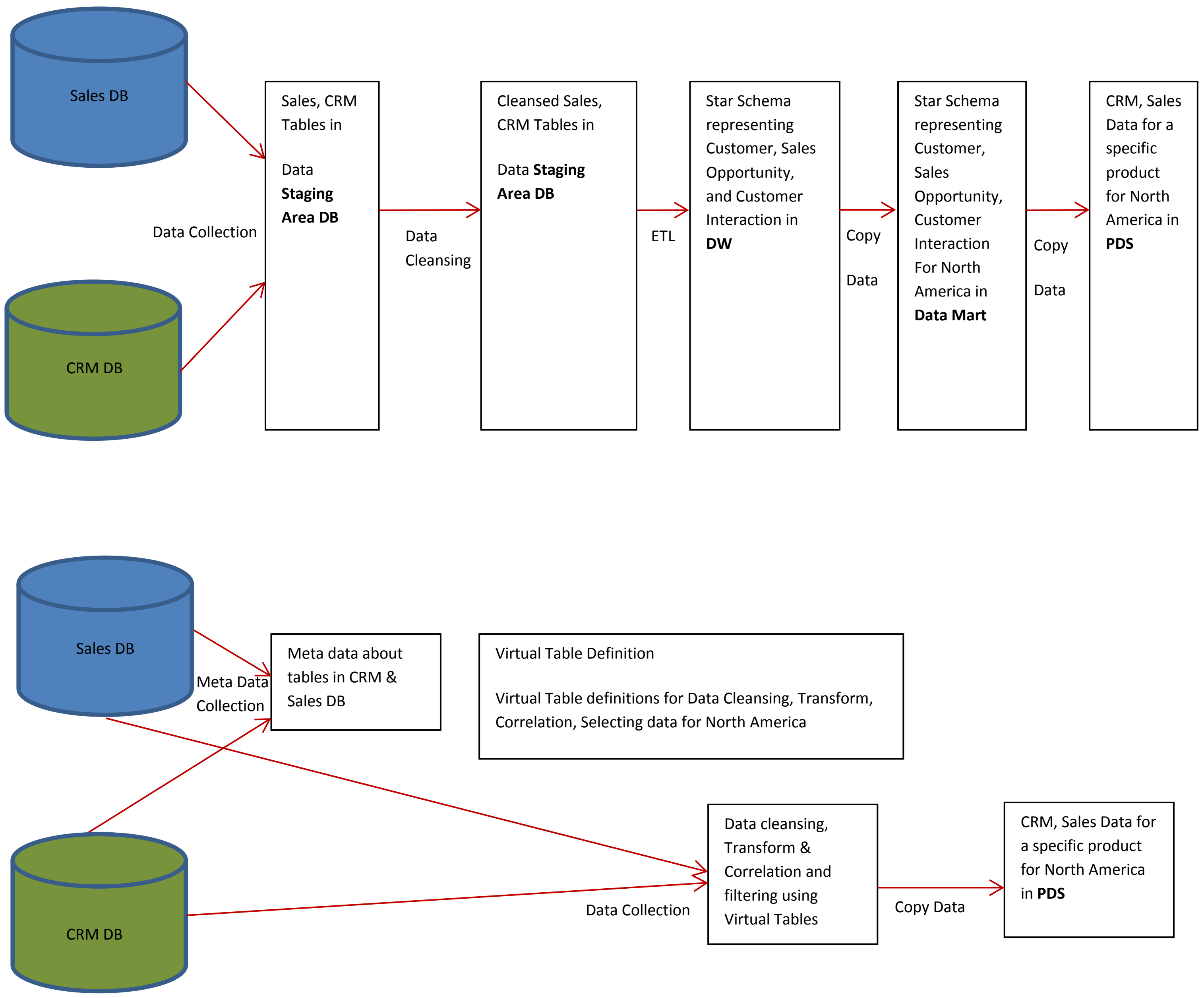
6.1 Comparison of Form Centric Architecture

In order to high light differences in system architecture between traditional data integration using data ware housing and Data Virtualization we employ form centric architectural views of the systems. Rectangular boxes represent the form and oval boxes call out the process being applied on the form.



System Architecture Comparison – Form Centric View
Figure: 10

Following example depicts DV vs. DW architectural differences in form centric view. Assume that we want to generate a report that will show light on cross selling opportunity; i.e selling more products and services to existing customers. To generate this report let's assume that we need to analyze data from CRM systems with sales systems. In the traditional BI approach data from sales and CRM system will be copied on to data staging area tables. Data cleansing is then performed on the data from staging area and written back to staging area tables. ETL is then employed to copy this data in to DW. A sub set of data is then copied over to Data Mart. Data from Data Mart is then copied over to PDS for analysis. In DV, the metadata about tables present in each of CRM and Sales database is found out. Virtual Tables are then defined to extract, transform, correlate and analyze data. Data gets copied over from production systems to PDS directly after going through data analysis pipeline.


Sales DB
CRM DB
Data Collection
Sales, CRM Tables in Data **Staging Area DB**
Data Cleansing
Cleansed Sales, CRM Tables in Data **Staging Area DB**
ETL
Star Schema representing Customer, Sales Opportunity, and Customer Interaction in **DW**
Copy Data
Star Schema representing Customer, Sales Opportunity, Customer Interaction For North America in **Data Mart**
Copy Data
CRM, Sales Data for a specific product for North America in **PDS**
Sales DB
CRM DB
Meta Data Collection
Meta data about tables in CRM & Sales DB
Virtual Table Definition
Virtual Table definitions for Data Cleansing, Transform, Correlation, Selecting data for North America
Data Collection
Data cleansing, Transform & Correlation and filtering using Virtual Tables
Copy Data
CRM, Sales Data for a specific product for North America in **PDS**

System Architecture Comparison – Example of Form Centric View, ETL/DW vs. DV
Figure: 11

The complexity of the system architecture is a measure of structural elements of the system and interactions among the elements. By using the above form centric architectural view, the complexity measure of the system can be broadly represented by the following:

Complexity of Traditional Data Integration = f3(f1(6), f2(5))

Where 6 is the number of form and 5 is the number of process in traditional data integration; f1, f2 and f3 are functions representing some measure of complexity given the number of form and process.

Complexity of Data Virtualization = f3(f1(4), f2(3))

Where 4 is the number of form and 3 is the number of process in DV; f1, f2 and f3 are functions representing some measure of complexity given the number of form and process.

Since functions f1, f2 and f3 has to be additive or multiplicative in nature, we can safely conclude that "Complexity of Traditional Data Integration" > "Complexity of Data Virtualization".

6.2 Comparison of Dependency Structure Matrix

Dependency Structure Matrix (DSM) capturing the structural and process elements of traditional data integration is given below. We have only captured high level elements and excluded elements that are common to data warehouse approach and data virtualization approach. Matrix is more detailed than from centric view of system architecture. Complexity of system can be measured by dimensions of the matrix and dependencies between various elements within the matrix.

DSM matrix for traditional data integration has a dimension of 35x35. The dependency among various elements in the matrix is 68. Complexity of Traditional Data Integration = f3 (f1(35,35), f2(68))

DSM matrix for traditional data integration with enterprise data standardization has a dimension of 33x33. The dependency among various elements in the matrix is 62. Complexity of Traditional Data Integration with enterprise data standardization = f3 (f1(33,33), f2(62))

DSM matrix for Data Virtualization has a dimension of 11x11. The dependency among various elements in the matrix is 13. Complexity of Data Virtualization = f3 (f1(11, 11), f2(13))

DSM matrix for Data Virtualization with enterprise data standardization has a dimension of 9x9. The dependency among various elements in the matrix is 11. Complexity of Data Virtualization = f3 (f1(9, 9), f2(11))

The function f1, f2, f3 is either additive or multiplicative in nature; hence from above we can conclude that :

"Complexity of Traditional Data Integration" > "Complexity of Traditional Data Integration with Enterprise Data Standardization" > "Complexity of Data Virtualization" > "Complexity of Data Virtualization with Enterprise Data Standardization".

From DSM analysis of system architecture we can conclude that "Data Virtualization" and "Enterprise Data Standardization" reduces system complexity considerably.

6.2.1 DSM for Traditional Data Integration

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
1	Data Sources	X																																		
2	Data collection tools	D	X																																	
3	Schedule Data Collection	D	D	X																																
4	Collect Data from data Sources	D	D	D	X																															
5	Work Flow to procure staging area DB					X																														
6	Data Staging Area Database	D			D		X																													
7	Schemas of data staging area	D			D		D	X																												
8	Write data to data staging area	D					D	D	X																											
9	Staging Area DB Maintenance			D			D		D	X							D						D													
10	Work flow to procure data error detec										X																									

	tion tools																																			
11	Tools for data error detection				D			D			D	X																								
12	Error detection in data				D							D	X																							
13	Data correction for data entry errors				D							D	D	X																						
14	Detect semantic differences in data				D			D				D			X																					
15	Transformations for semantic errors							D							D	X																				
16	Writing modified data back in to data staging area							D						D		D	X																			
17	Work flow for procurement of ETL tool																	X																		
18	ETL Tool						D	D										D	X	D	D						D									
19	Data Joining																		D	X								D								
20	Data Transformation for analysis																		D		X							D								
21	Data Loading in																		D			X					D									

	to Data warehouse																																			
22	Schedule for ETL																D						X				D									
23	Data deletion from staging area																						D	X												
24	ETL tool Maintenance																						D	D	X											
25	Work flow to procure Data warehouse DB																									X										
26	Data Warehouse DB																									D	X	D								
27	Data Warehouse Schema							D																				X						D		
28	Policies for managing historical data																						D						X		D					
29	Data Warehouse maintenance																						D						D	X	D					
30	Data Archival Mechanisms																												D	D	X					
31	Workflow to procure Data Mart DB																															X				
32	Data Mart DB																															D	X			
33	Data Mart																											D					D	X		

	Schema																																			
34	Schedule for data copy from data warehouse to data mart																					D											X	
35	Data Mart DB Maintenance																																	D	X

🟦 Data Collection 🟫 Data Staging 🟨 Data Cleansing 🟩 ETL 🟪 Data Warehouse 🟧 Data Mart D - Dependency

DSM for Traditional BI
Figure: 12

6.2.2 DSM for Data Virtualization

		1	2	3	4	5	6	7	8	9	10	11
1	Data Sources	X										
2	Work Flow to procure Data Virtualization (DV) Software		X									
3	DV connecting to data sources	D	D	X								
4	DV introspecting data sources & capturing metadata	D		D	X							
5	Maintenance of DV software			D	D	X						
6	View definition to check for data errors and to correct data errors	D					X					
7	Detect schema difference automatically using DV software				D			X				
8	View definitions for Transformations to handle semantic errors							D	X			
9	Views defining Data Joins	D								X	D	
10	Views Defining Data Transformation for analysis									D	X	D
11	Views defining data transformation for data analysis application specific formats											X

🟦 Data Collection Setup 🟨 Data cleansing Views 🟩 Join & Transformation Views D - Dependency

DSM for Data Virtualization Based BI
Figure: 13

6.2.3 DSM for Traditional Data Integration with Enterprise Data Standardization

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
1	Data Sources	X																																
2	Data collection tools	D	X																															
3	Schedule Data Collection	D	D	X																														
4	Collect Data from data Sources	D	D	D	X																													
5	Work Flow to procure staging area DB					X																												
6	Data Staging Area Database	D			D		X																											
7	Schemas of data staging area	D			D		D	X																										
8	Write data to data staging area	D					D	D	X																									
9	Staging Area DB Maintenance			D			D		D	X					D						D													
10	Work flow to procure data error detection tools										X																							
11	Tools for data error detection				D			D			D	X																						
12	Error detection in data				D							D	X																					
13	Data correction for data entry errors				D							D	D	X																				
14	Writing modified data back in to data staging area							D						D	X																			
15	Work flow for procurement of ETL tool															X																		
16	ETL Tool						D	D								D	X	D	D						D									
17	Data Joining																D	X								D								
18	Data Transformation for analysis																D		X							D								
19	Data Loading in to Data Ware House																D			X					D									
20	Schedule for ETL														D						X				D									
21	Data deletion from staging area																				D	X												
22	ETL tool Maintenance																				D	D	X											
23	Work flow to procure Data warehouse DB																							X										
24	Data warehouse DB																							D	X	D								
25	Data warehouse Schema							D																		X						D		
26	Policies for managing historical data																				D						X		D					
27	Data warehouse maintenance																				D						D	X	D					
28	Data Archival Mechanisms																										D	D	X					
29	Workflow to procure Data Mart DB																													X				
30	Data Mart DB																													D	X			
31	Data Mart Schema																									D					D	X		
32	Schedule for data copy from data ware house to data mart																				D												X	
33	Data Mart DB Maintenance																																D	X

🟦 Data Collection 🟥 Data Staging 🟨 Data Cleansing 🟩 ETL 🟪 Data Warehouse 🟧 Data Mart D - Dependency

DSM for Traditional BI with Enterprise Data Standardization
Figure: 14

6.2.4 DSM for Data Virtualization with Enterprise Data Standardization

		1	2	3	4	5	6	7	8	9
1	Data Sources	X								
2	Work Flow to procure Data Virtualization (DV) Software		X							
3	DV connecting to data sources	D	D	X						
4	DV introspecting data sources & capturing metadata	D		D	X					
5	Maintenance of DV software			D	D	X				
6	View definition to check for data errors and to correct data errors	D					X			
7	Views defining Data Joins	D						X	D	
8	Views Defining Data Transformation for analysis							D	X	D
9	Views defining data transformation for data analysis application specific formats									X

🟦 Data Collection Setup 🟨 Data cleansing Views 🟩 Join & Transformation Views D - Dependency

DSM for DV based BI with Enterprise Data Standardization
Figure: 15

Chapter 7

System Dynamics Model Comparison

Our research has indicated that there are two primary benefits assigned to Data Virtualization technology over Data Ware Housing:

1. Lower Operational Cost
2. Shorter Time-To-Market

Both of these attributes then in turn had transitive effects in that with the same IT budget now Business Analysts could do lot more analysis. Effectively Data Virtualization offered lower cost per experiment/analysis. We employ System dynamics model to check if these benefits are feasible or not.

Systems dynamics is a methodology and modeling technique often employed to understand the dynamics of complex systems. System Dynamics took root in 1950's, developed by Professor Jay Forrester of Massachusetts Institute of Technology. According to System dynamics theory, systems have structure which is composed of circular, interlocking and often time delayed relationship among its components. System Dynamics model consists of four principle elements:

- Feedback Loops
  In many systems, information/effects resulting from actions of system components would return back to the origin of action in some other form. For example in under developed countries that doesn't offer much social security, people have more children as a way to increase their income and as a form of retirement security; as number of people in the country increases, the resource contention increases, which then reduces income potential of individuals, this forces more people to depend on the meager social security which would go down as a result of more people depending on it, this in turn causes population to increase as people would have more kids to improve their revenue.

  Feedback loops could be balancing or reinforcing loop. Example described above is a case of reinforcing loop. In balancing loop system would end up in a balance or stable state. Demand and Supply is an example of balancing loop; When demand goes up, price of the commodity would go up which would then incentivize more people to supply/produce/grow the commodity. As a result there would be more availability of the commodity which results in price to go down which in turn would de-incentivize people from supplying/producing/growing the commodity.

- Flows
  Flows as the name suggests is the flow of information or physical entity.
  New births are an example of flow. Flow could be inflow or outflow. For example new birth is an inflow to population and death is an outflow from population.

- Stock
  Stock is the accumulation of flow. According to Principle of accumulation all dynamic behavior in the world occurs when flows accumulate in stocks. New births are flow in to stock of population.

- Time Delays
  There could be time delays in flows of the system. One cause for this time delay is the difference between inflow and outflow rates. For example consider fossil fuel burning and global temperature; even though fossil fuel contributes directly to global warming, the effect is not felt till some time in future.

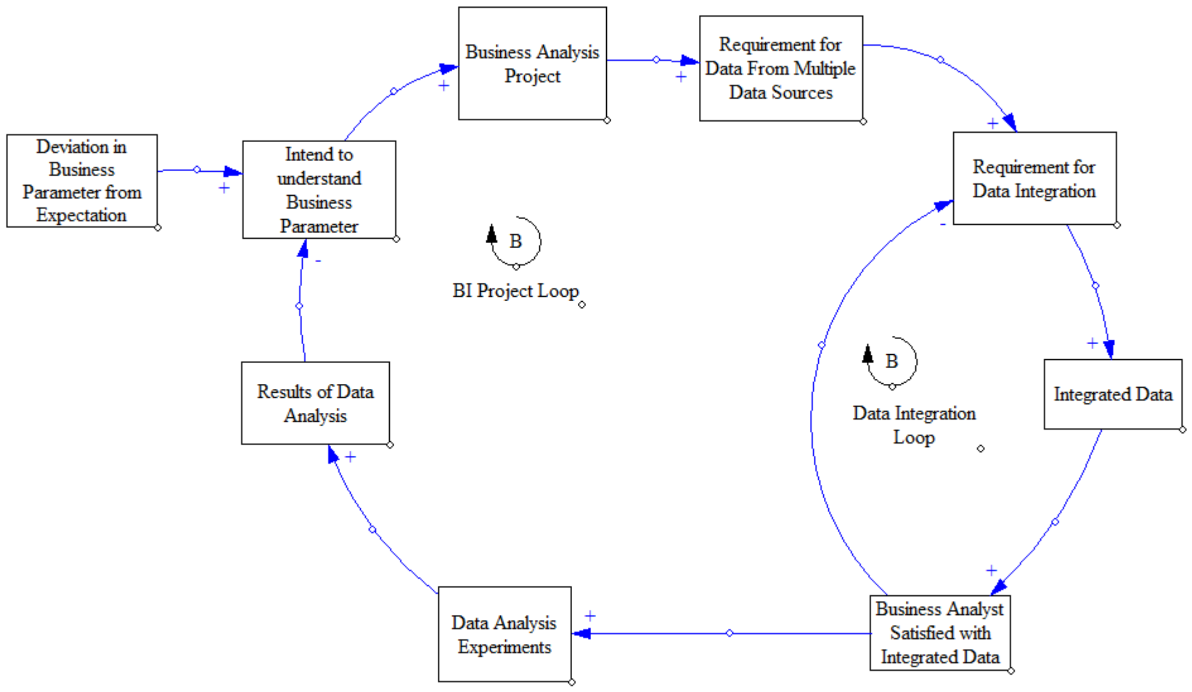
System Dynamics uses two principle models to represent two different aspects of system.

- Causal Loop
  This model is built primarily to explain what causes the effects that are seen on a system.

- Stock and Flow Diagrams
  These models are built to explore dynamics of the system. Such models would contain stocks and flows, and would have equations defined. Simulating these models would help us measure the dependencies among various elements of the system.

7.1 What drives Data Integration?

Here we employ system dynamics causal loop diagram to explain the drivers for Data Integration. Most often BI projects start out with a variation observed in business performance parameter. To understand this variation a BI project is initiated; this then requires data integration. As the project succeeds the variation is business parameter is understood which then reduces the "intend to understand business". This forms a balancing loop.


Deviation in Business Parameter from Expectation
Intend to understand Business Parameter
Business Analysis Project
Requirement for Data From Multiple Data Sources
Requirement for Data Integration
Integrated Data
Business Analyst Satisfied with Integrated Data
Data Analysis Experiments
Results of Data Analysis
B
BI Project Loop
B
Data Integration Loop
+
-

Data Integration Driver

Figure: 16

7.2 Operational Cost comparison

We employ System Dynamics Stock & Flow model to understand the operational cost for DW and DV.

The traditional DW approach to data integration is broken in to seven different phases:

- Data Capture
- Data in Staging Area
- Data Cleansing
- ETL (Data Join, Transform & load in to DW)
- Copy to Data Mart
- Copy Data to PDS
- Potential Changes to artifacts in all the above phases due to changes in requirements

DV on the other hand has fewer phases as it has fewer moving parts. The phases of data integration using DV can be broken in to three phases:

- Data Capture
- Data Cleansing, Data Join, Transform using DV
- Data Caching
- Copy Data to PDS

Requirement changes in case of DV wouldn't require schema changes and can be handled by modifying the views.

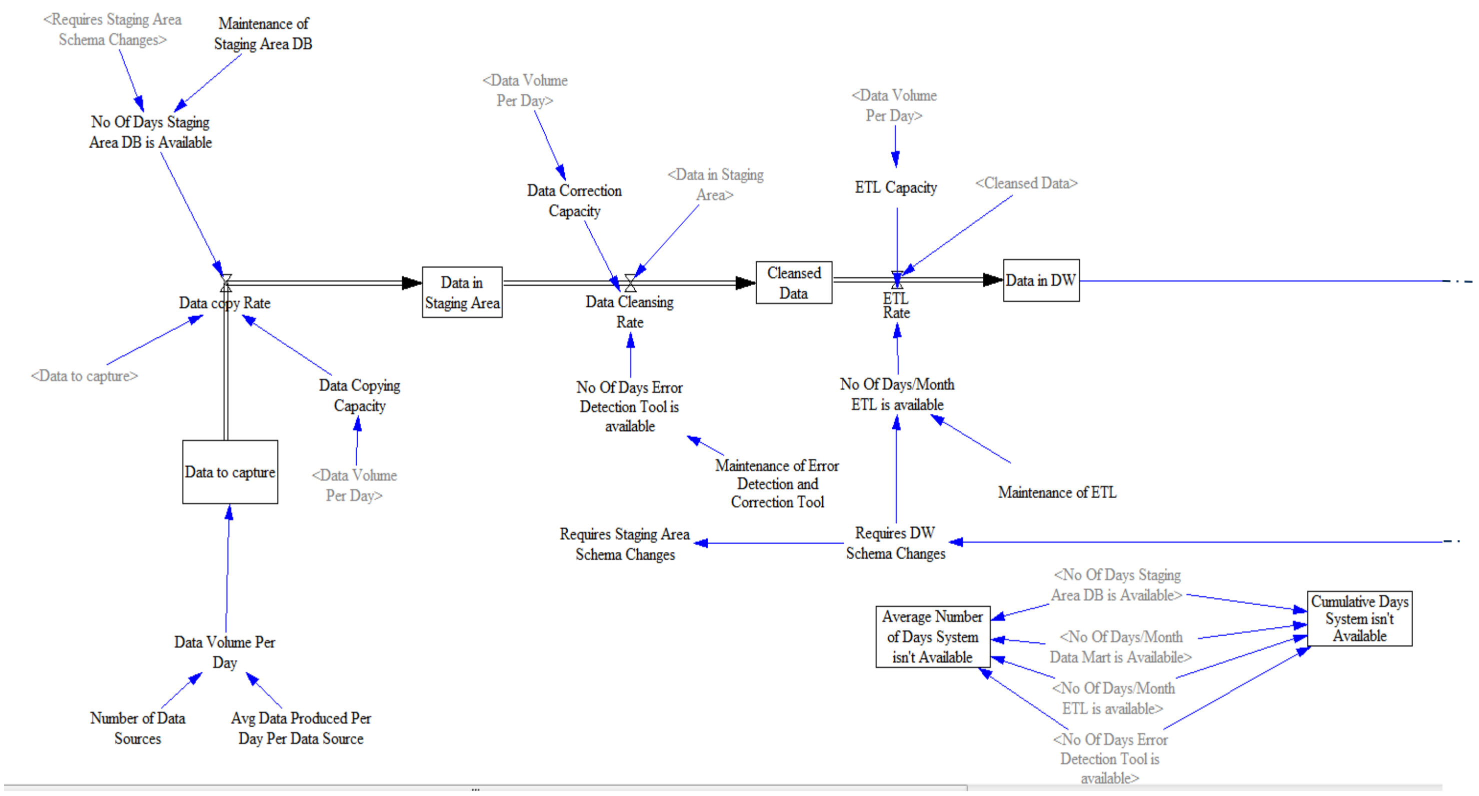
An important difference between two architectures is the difference in number of moving parts between the two. This difference in the number of moving parts affects two aspects of operational cost:

- System UP Time
  As the number of moving part increases, the system failure rate increases. Our simulation shows on average DV cuts system down time by more than half compared to DW approach. On the other hand DV becomes single point of failure for the whole system. If DV is down then none of the analysis can proceed. In the traditional BI approach if ETL is down, analysis can still continue on old data kept in DW.

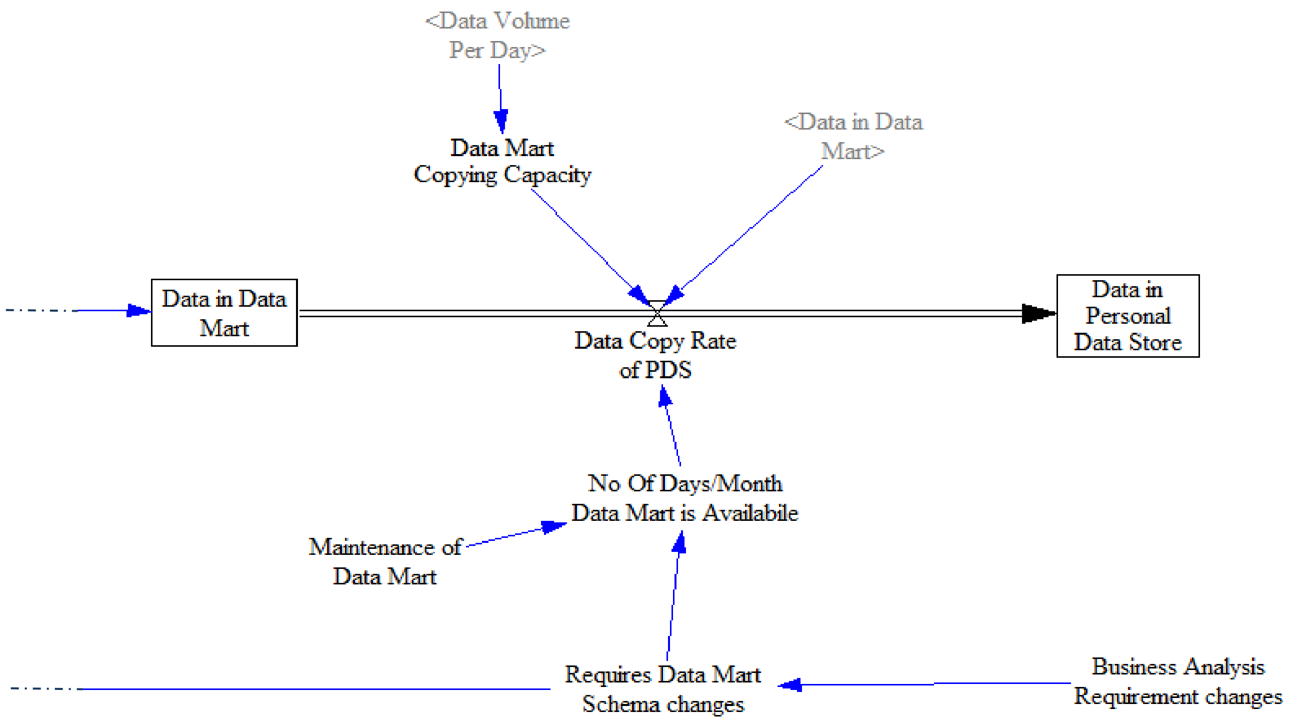
- Operational Cost
  Each of the data base and software tools requires maintenance. Our simulation shows DW approach requiring around 100 days' worth of maintenance to fix the system. DV on the other hand requires around 20 days' worth of fixing. Add to this preventive maintenance, like data archival needed for data bases to free up the space. Please note that for most system elements we expect the system failure would follow normal distribution between 0 days to 2 days with a mean of 1 day and standard deviation of 0.5 days each month.

Comparison of Stock and Flow models clearly shows DV to be much better solution in terms of operational cost.

7.2.1 DW - Operational Dynamics Model



<Requires Staging Area Schema Changes>
Maintenance of Staging Area DB
No Of Days Staging Area DB is Available
<Data Volume Per Day>
Data Correction Capacity
<Data in Staging Area>
<Data Volume Per Day>
ETL Capacity
<Cleansed Data>
Data copy Rate
Data in Staging Area
Data Cleansing Rate
Cleansed Data
ETL Rate
Data in DW
<Data to capture>
Data Copying Capacity
No Of Days Error Detection Tool is available
No Of Days/Month ETL is available
Data to capture
<Data Volume Per Day>
Maintenance of Error Detection and Correction Tool
Maintenance of ETL
Requires Staging Area Schema Changes
Requires DW Schema Changes
<No Of Days Staging Area DB is Available>
Average Number of Days System isn't Available
<No Of Days/Month Data Mart is Availabile>
Cumulative Days System isn't Available
<No Of Days/Month ETL is available>
<No Of Days Error Detection Tool is available>
Data Volume Per Day
Number of Data Sources
Avg Data Produced Per Day Per Data Source



<Data Volume Per Day>
Data Mart Copying Capacity
<Data in Data Mart>
Data in Data Mart
Data Copy Rate of PDS
Data in Personal Data Store
No Of Days/Month Data Mart is Availabile
Maintenance of Data Mart
Requires Data Mart Schema changes
Business Analysis Requirement changes

DW – Operational Dynamics

Figure: 17

Overview of Stock and Flow model for DW Operational Dynamics is as follows:

- Data produced by "data sources" (production systems) needs to be copied for analysis each day.
- Size of data that needs to be copied is represented by stock "Data to capture" which depends on "Number of Data Sources" and "Avg Data produced Per day per data source".
- Based on "Data copy rate" data flows out from "Data to capture" stock to "Data in Staging Area" stock. "Data copy rate" depends on availability of "Staging Area DB". Preventive maintenance and "Staging area schema changes" could make "Staging Area DB un available".
- Based on "Data Cleansing Rate" data flows out from "Data in Staging Area" stock to "Cleansed Data" stock.
- We assume that "Data Cleansing Rate" can match "Data Copy Rate". "Data Cleansing Rate" would go down when "Staging area DB" is down for maintenance or for schema changes or for maintenance of data correction tool.
- Data from "Cleansed Data" stock flows in to DW based on ETL rate.
- We assume "ETL Rate" can match "data copy rate". "ETL Rate" would go down if "Staging area DB" is down, or if ETL tool is down for maintenance or if DW is down.
- Data from "DW" stock flows in to "Data Mart" stock.
- Data from "Data Mart" stock then flows in to "PDS" based on "Data Copy Rate of PDS".
- We assume "Data Copy Rate of PDS" would only go down if Data Mart is down either for maintenance or for schema changes.

Key aspects of Stock and Flow model for DW Operational Dynamics are as follows:

- Number of Data Sources from which data needs to be integrated is set as 10.

- Data Copying/cleansing/ETL capacity is set to be same as Data Generated each day, thus ignoring artificial bottlenecks

- Number of days Staging Area DB/ETL-DW/Data Mart is unavailable is set to follow RANDOM NORMAL(0 , 2 , 1 , 0.5 , 0); i.e. minimum possible value of 0, maximum of 2, average of 1 with std. deviation of 0.5 days. Note that each simulation cycle is a month; so we expect on average 1 day of failure out of 30.

- Number of days error detection tool is unavailable is set to follow RANDOM NORMAL(0 , 2 , 1.5 , 0.5 , 0); i.e. minimum possible value of 0, maximum of 2, average of 1.5 with std. deviation of 0.5 days. Since data entry errors can be random and can be more frequent we increased the average failure time to 1.5 days to accommodate for new types of data entry errors. Note that each simulation cycle is a month; so we expect on average 1.5 day of failure out of 30.

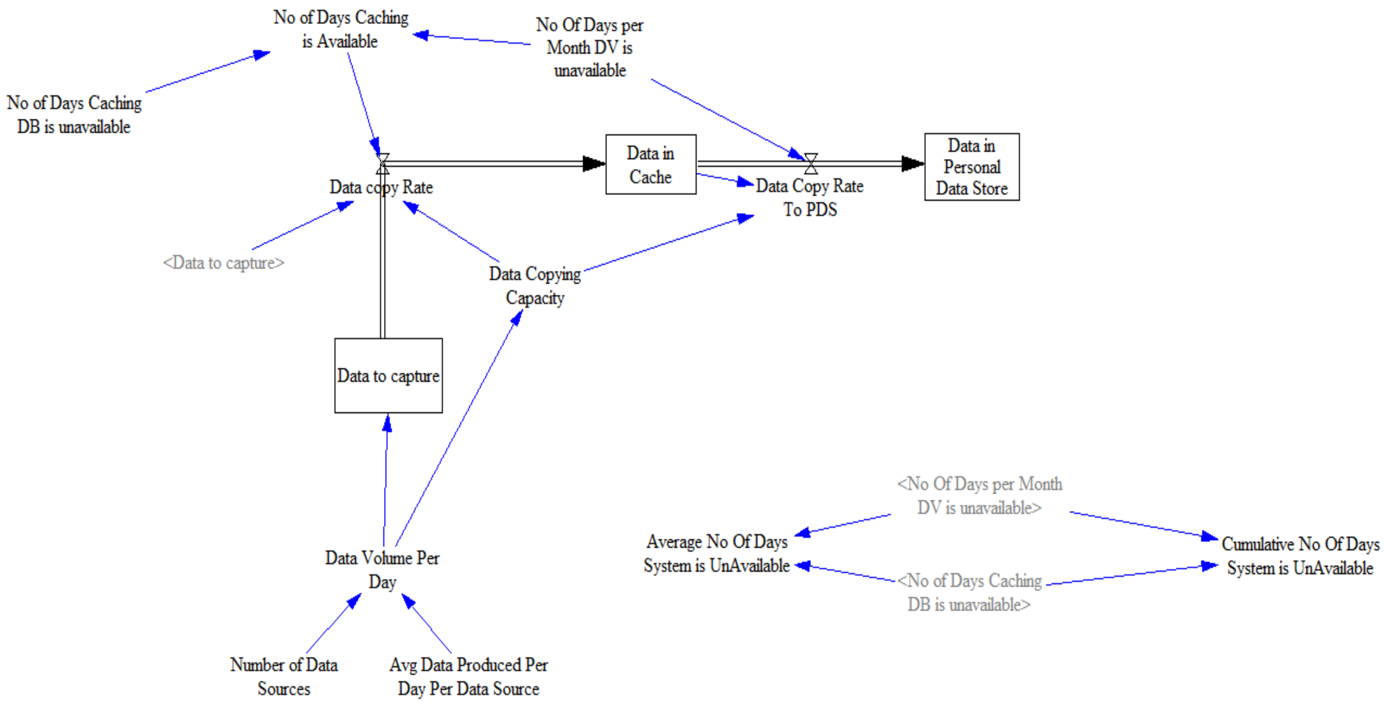
- We assume business requirement changes would follow RANDOM NORMAL(0 , 3 , 1.5 , 0.5 , 0); i.e. on average every month there would be 1.5 days would be spend on analyzing data integration changes needed to satisfy the business requirement. Data integration requirement changes would also result in potential schema changes all along the data flow pipe line.

- We compute Cumulative Number of Days System Unavailable as = No of Days/Month Staging Area DB is unavailable + No of Days/Month ETL/DW is unavailable + No of Days/Month error detection tool is unavailable + No of Days/Month Data Mart is unavailable.

- We compute Average Number of Days System unavailable as = max(No of Days/Month Staging Area DB is unavailable, No of Days/Month ETL/DW is unavailable, No of Days/Month error detection tool is unavailable, No of Days/Month Data Mart is unavailable). We use the max value because we assume the individual component failures would overlap.

- We used 1 month period and used 12 such periods for model simulation.

7.2.2 DV - Operational Dynamics Model


No of Days Caching is Available
No Of Days per Month DV is unavailable
No of Days Caching DB is unavailable
Data in Cache
Data in Personal Data Store
Data copy Rate
Data Copy Rate To PDS
<Data to capture>
Data Copying Capacity
Data to capture
<No Of Days per Month DV is unavailable>
Average No Of Days System is UnAvailable
Cumulative No Of Days System is UnAvailable
<No of Days Caching DB is unavailable>
Data Volume Per Day
Number of Data Sources
Avg Data Produced Per Day Per Data Source

DV – Operational Dynamics

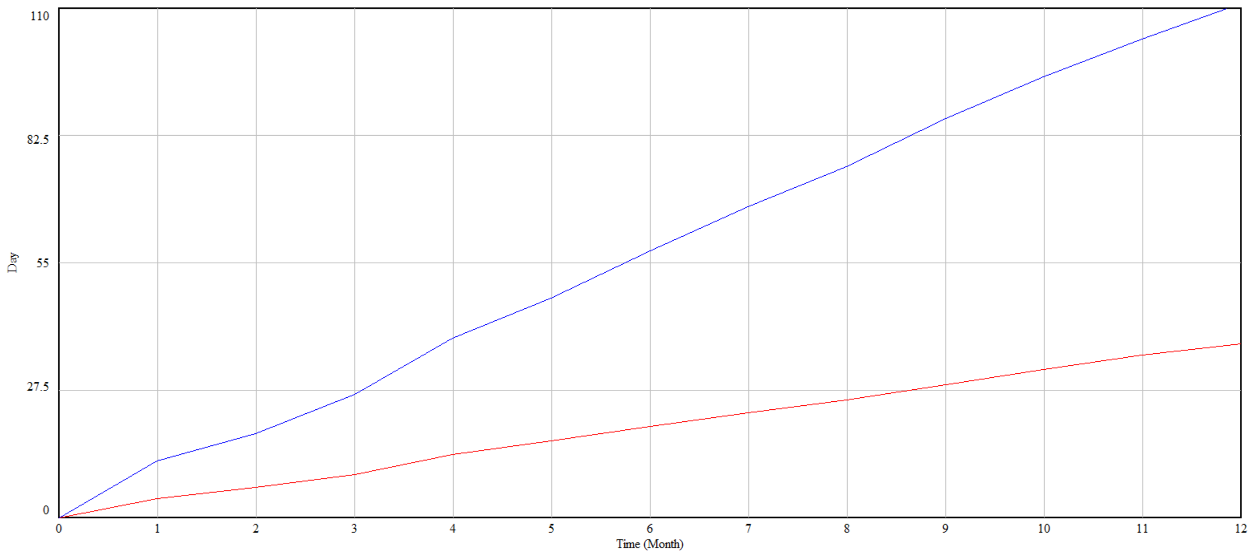
Figure: 18

Overview of Stock and Flow model for DV Operational Dynamics is as follows:

- Data produced by "data sources" (production systems) needs to be copied for analysis each day.
- Size of data that needs to be copied is represented by stock "Data to capture" which depends on "Number of Data Sources" and "Avg Data produced Per day per data source".
- Based on "Data copy rate" data flows out from "Data to capture" stock to "Data in Cache" stock. "Data copy rate" depends on availability of "Caching DB" and “DV”. Note here that for system dynamics model we are assuming that DV deploys a Cache DB to store data from production systems.
- Data from “Cache DB” flows out to PDS based on “Data Copy Rate To PDS”. Availability of DV and Cache DB affects “Data Copy Rate To PDS”.

Key aspects of Stock and Flow model for DV Operational Dynamics are as follows:

- Number of Data Sources from which data needs to be integrated is set as 10.
- Data Copying/Caching capacity is set to be same as Data Generated each day, thus ignoring artificial bottlenecks
- Number of days Caching DB/DV is unavailable set to follow RANDOM NORMAL(0 , 2 , 1 , 0.5 , 0); i.e. minimum possible value of 0, maximum of 2, average of 1 with std. deviation of 0.5 days. Note that each simulation cycle is a month; so we expect on average 1 day of failure out of 30.
- We compute Cumulative Number of Days System unavailable as = No of Days/Month DV is unavailable + No of Days/Month caching DB is unavailable.
- We compute Average Number of Days System unavailable as = max(No of Days/Month DV is unavailable, No of Days/Month caching DB is unavailable). We use the max value because we assume the individual component failures would overlap.
- We used 1 month period and used 12 such periods for model simulation.

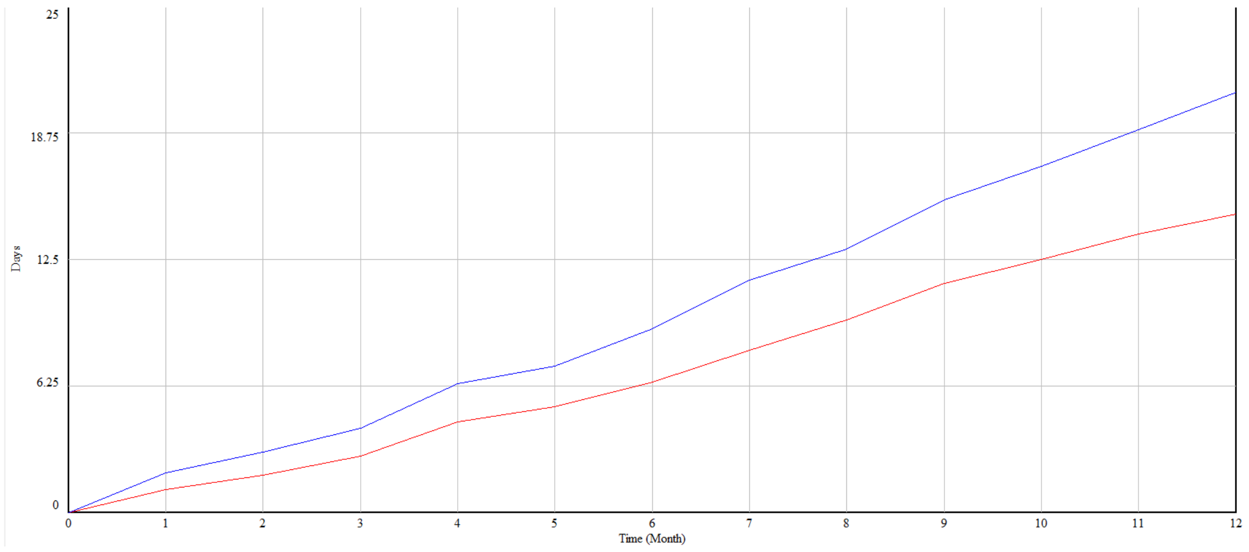


——— Cumulative Number of Days ETL-DW System is unavailable/Year

——— Average Number of Days ETL-DW System is unavailable/Year

DW System Un- Availability per Year

Figure: 19



—— Cumulative Number of Days DV System is unavailable/Year

—— Average Number of Days DV System is unavailable/Year

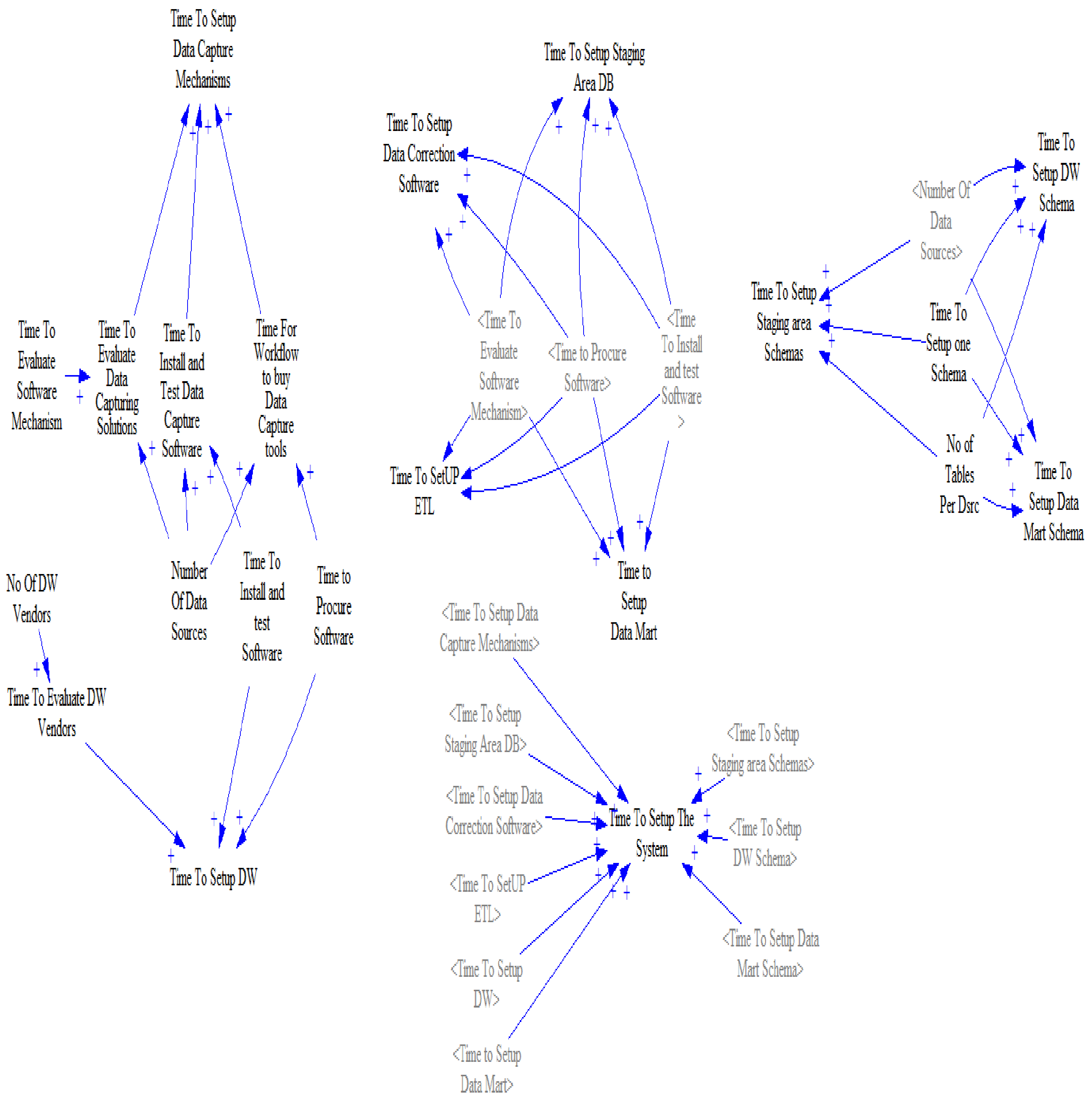
DV System Un- Availability per Year

Figure: 20

7.3 Time-To-Market Comparisons

Here we employ System Dynamics Stock & Flow model to understand the Time it takes to complete the project using DW and DV. We computed time to implement data integration solution using traditional DW approach and new DV method. Our model simulation shows on average DV solution takes less than 50% of time taken to implement DW solution. This difference can be attributed to the less number of parts involved in DV comparing to DW. Each piece of software requires, vendor evaluation and selection, negotiations about financial and legal aspects of the deal, work flow to approve the purchase, installing and testing the software and finally customizing the software for the specific need.

7.3.1 DW – Time-To-Market Model



Time To Setup Data Capture Mechanisms
Time To Setup Staging Area DB
Time To Setup Data Correction Software
Time To Setup DW Schema
<Number Of Data Sources>
Time To Setup Staging area Schemas
Time To Setup one Schema
No of Tables Per Dsrc
Time To Setup Data Mart Schema
Time To Evaluate Software Mechanism
Time To Evaluate Data Capturing Solutions
Time To Install and Test Data Capture Software
Time For Workflow to buy Data Capture tools
<Time To Evaluate Software Mechanism>
<Time to Procure Software>
<Time To Install and test Software>
Time To SetUP ETL
Time to Setup Data Mart
No Of DW Vendors
Number Of Data Sources
Time To Install and test Software
Time to Procure Software
Time To Evaluate DW Vendors
Time To Setup DW
<Time To Setup Data Capture Mechanisms>
<Time To Setup Staging Area DB>
<Time To Setup Staging area Schemas>
<Time To Setup Data Correction Software>
Time To Setup The System
<Time To Setup DW Schema>
<Time To SetUP ETL>
<Time To Setup Data Mart Schema>
<Time To Setup DW>
<Time to Setup Data Mart>

DW System Time-To-Market Model

Figure: 21

Key aspects of Stock and Flow model for DW Time-To-Market model are as follows:

- Traditional data integration is broken down in to nine distinct steps. Six to select, install and test software and three steps to customize it for data integration needs. Time to implement DW is the sum of time taken to implement each of these phases. The six software setup steps are
  - Data Capturing Software: This can be CDC (Change Data Capture), replication or just querying data for changes.
  - Staging Area DB
  - Data Error Detection Software
  - ETL Software
  - Data warehouse: We assume there are only 3 principal vendors here (Teradata, IBM, Oracle)
  - Data Mart

  Three steps to customize software are:

  - Setup staging area schemas: We assume staging area schemas would mimic data source schemas. For the purpose of simulation we assumed there to be 10 data sources each one containing 4 different tables.
  - Setup DW schemas: For simulation we assumed DW would have same number of tables as Data Staging Area. In Star/Snowflake schema this would tend to be more or less true.
  - Setup Data Mart Schemas: For simulation we assumed Data mart would have same number of tables as Data Staging Area.

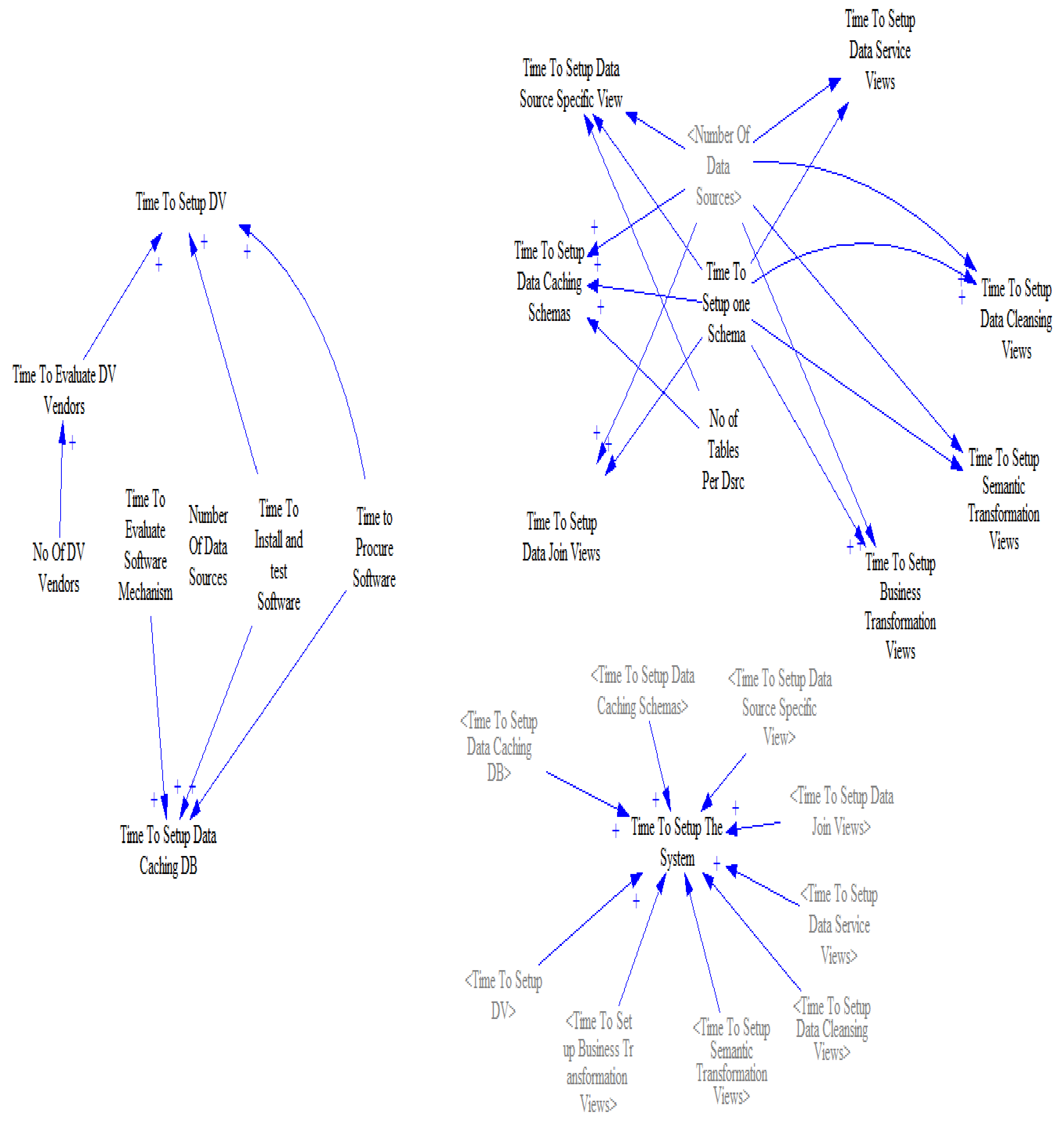
- We assumed software evaluation to follow RANDOM NORMAL(0, 5, 2, 1, 0); i.e it will take on average 2 days to evaluate a piece of software with a std. deviation of 1 day and max possible value of 5 days.

- We assumed software installation and testing to follow RANDOM NORMAL(0, 5, 2, 0.5, 0); i.e it will take on average 2 days to install and test a piece of software with a std. deviation of 0.5 day and max possible value of 5 days.

- We assumed software procurement process to follow RANDOM NORMAL(0, 5, 2, 0.5, 0); i.e it will take on average 2 days for work flow to complete to buy a piece of software with a std. deviation of 0.5 day and max possible value of 5 days. In most cases this is over optimistic.

- We run the simulation 100 times to find average time-to-market. Average time-to-market for ETL-DW solution is around 100 days.

7.3.2 DV – Time-To-Market Model


Time To Setup DV
Time To Evaluate DV Vendors
No Of DV Vendors
Time To Evaluate Software Mechanism
Number Of Data Sources
Time To Install and test Software
Time to Procure Software
Time To Setup Data Caching DB
Time To Setup Data Source Specific View
Time To Setup Data Service Views
<Number Of Data Sources>
Time To Setup Data Caching Schemas
Time To Setup one Schema
Time To Setup Data Cleansing Views
No of Tables Per Dsrc
Time To Setup Data Join Views
Time To Setup Semantic Transformation Views
Time To Setup Business Transformation Views
<Time To Setup Data Caching Schemas>
<Time To Setup Data Source Specific View>
<Time To Setup Data Caching DB>
Time To Setup The System
<Time To Setup Data Join Views>
<Time To Setup Data Service Views>
<Time To Setup DV>
<Time To Set up Business Tr ansformation Views>
<Time To Setup Semantic Transformation Views>
<Time To Setup Data Cleansing Views>

DV System Time-To-Market Model

Figure: 22

Key aspects of Stock and Flow model for DV Time-To-Market are as follows:

- DV data integration is broken down in to nine distinct steps. Time to implement DV is the sum of time taken to implement each of these phases. There are two steps to select, install and test software and seven steps to customize it for data integration needs. The two software setup steps are
    - DV Software: We assume there to be three DV vendors (Informatica, Composite Software, IBM)
    - Caching DB Software

  Seven steps to customize software are:

    - Setup caching area schemas: We assume caching DB schemas would mimic data source schemas. For the purpose of simulation we assumed there to be 10 data sources each one containing 4 different tables.
    - Data Source Specific views: For simulation we assumed DV would have same number of views as tables in all data sources; i.e. 40.
    - Data joining views: We assume there would be as many data join views as the number of data sources; i.e. 10. This is an overestimation; in practical usage the number of these views is much lower.
    - Data cleansing views: We assume there would be as many data cleansing views as the number of data sources; i.e. 10. This is an overestimation; in practical usage the number of these views is much lower.
    - Data service views: We assume there would be as many data service views as the number of data sources; i.e. 10. This is an overestimation; in practical usage the number of these views is much lower.
    - Semantic Transformation views: We assume there would be as many semantic transformation views as the number of data sources; i.e. 10. This is an overestimation; in practical usage the number of these views is much lower.
    - Business Transformation views: We assume there would be as many business transformation views as the number of data sources; i.e. 10. This is an overestimation; in practical usage the number of these views is much lower.

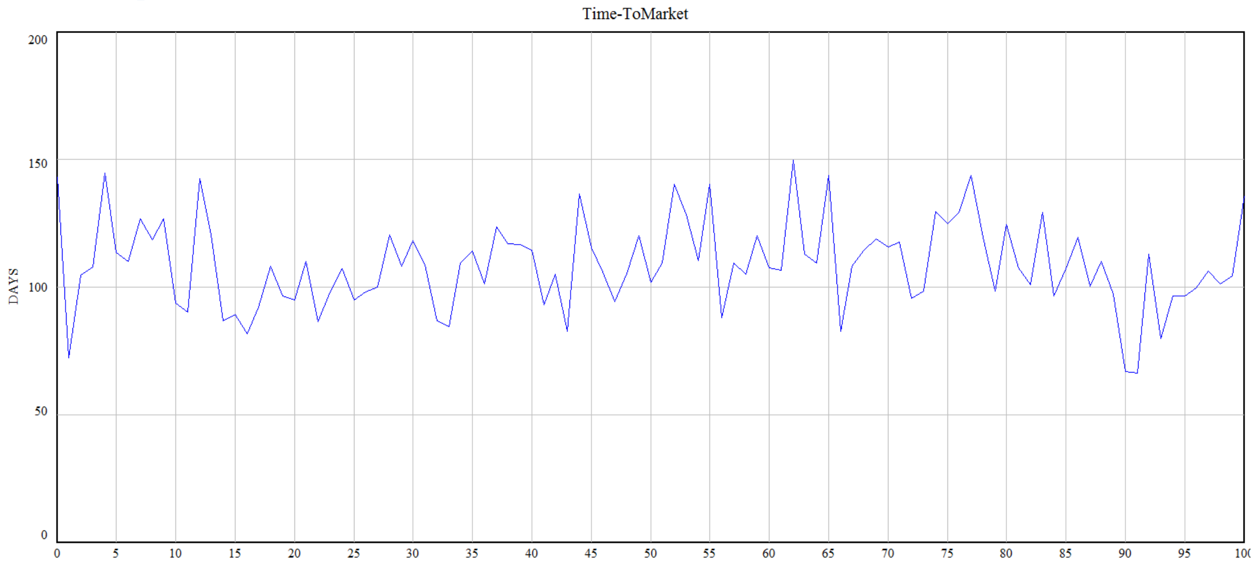
- We assumed software evaluation to follow RANDOM NORMAL(0, 5, 2, 1, 0); i.e it will take on average 2 days to evaluate a piece of software with a std. deviation of 1 day and max possible value of 5 days.

- We assumed software installation and testing to follow RANDOM NORMAL(0, 5, 2, 0.5, 0); i.e it will take on average 2 days to install and test a piece of software with a std. deviation of 0.5 day and max possible value of 5 days.

- We assumed software procurement process to follow RANDOM NORMAL(0, 5, 2, 0.5, 0); i.e it will take on average 2 days for work flow to complete to buy a piece of software with a std. deviation of 0.5 day and max possible value of 5 days. In most cases this is over optimistic.

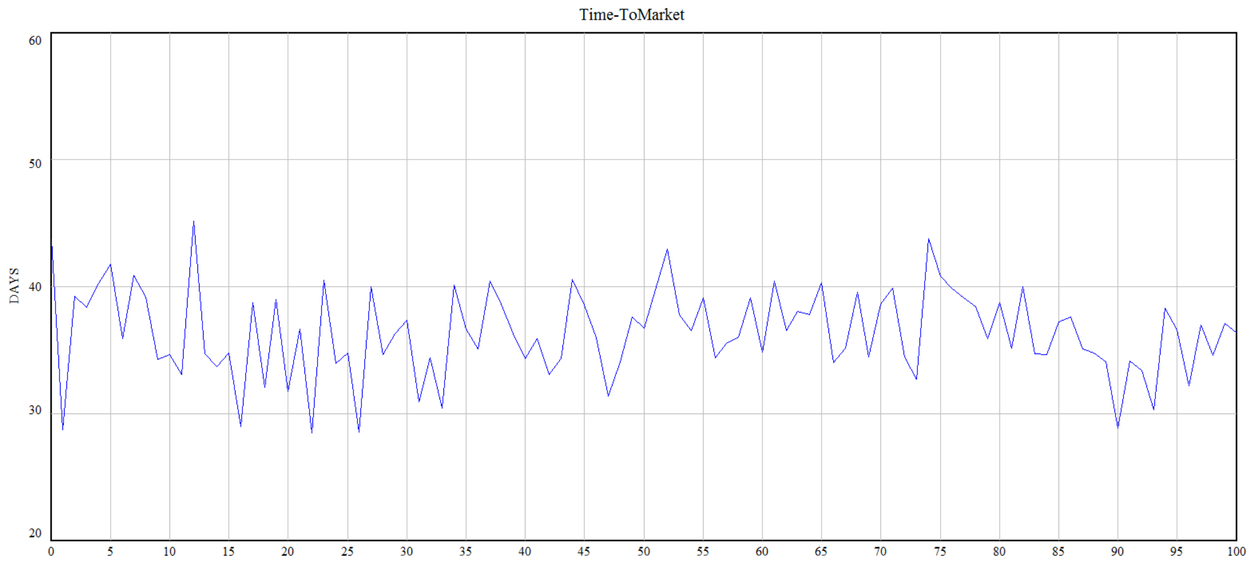
- We run the simulation 100 times to find average time-to-market. Average time-to-market for DV solution is around 37 days.

7.3.3 Comparison of DW & DV Time-To-Market



DW System - Time-To-Market

Figure: 23



DV System Time-To-Market

Figure: 24

Comparisons of system dynamics stock and flow models for ETL-DW and DV shows that DV has clear advantages over ETL-DW. Advantages of DV over ETL-DW are as follows:

- DV has better system uptime compared to ETL-DW solution. This is primarily because DV has less moving parts compared to ETL-DW.
- DV has shorter implementation cycle when compared to ETL-DW. This is primarily due to the fact that you need only one piece of software namely DV platform as opposed to numerous databases and schemas that one will have to define for ETL-DW. Also making changes to the system is lot easier in DV as opposed to in ETL-DW. In ETL-DW changes would often mean schema changes whereas in DV changes to virtual table can be achieved by changing SQL.
- DV is less costly compared to ETL-DW. In ETL-DW each DB requires, hardware and software maintenance; DBA's are needed to maintain DB whereas in DV one needs to maintain only DV software and cache DB.
- The above benefits of DV have other transitive benefits as well. Shorter implementation cycle implies project starts to produce results earlier and may save business additional dollars. For example assume that an analysis project for predicting customer churn came online 3 months earlier than ETL-DW would have taken. Because of DV, business may be able to save customers that would have left business otherwise.

Chapter 8

Case Studies

In this section we examine two implementations of data virtualizations; we look at the business problem and how data virtualization was employed to tackle the business problem. We then look at the benefits as perceived by the solution implementer. These case studies have been taken from the book "Data Virtualization", First Edition printed in September 2011, by Judith R Davis and Robert Eve.

8.1 NYSE Euronext

8.1.1 Organization Background

NYSE Euronext is comprised of six equity exchanges and eight derivatives exchanges. NYSE, NYSE Arca, NYSE Liffe, Euronext and AMEX are some of them. These are world's leading and most liquid equities and derivatives exchanges. Over 8,500 issues are listed on NYSE Euronext exchanges and exchange handles over four billion transactions per day with an average daily value of $153 billion.

8.1.2 Business Problem

NYSE Euronext's business brings with it many challenges for data integration. Some of these challenges are:

- NYSE Euronext has gone through multiple mergers and acquisitions over the years. Integrating data from these different exchanges poses significant complexity.
- NYSE Euronext trades 14 different types of products in its exchanges and each product has different data structure.
- Each day NYSE Euronext has to deal with massive data volumes, anywhere from 2TB to 5 TB.
- NYSE Euronext needs to meet strict SLA's for data delivery.

In addition to the above, NYSE Euronext has a new strategic initiative to provide a "Secure Financial Transaction Infrastructure" (SFTI) a SaaS offering designed to provide customers with professional services built around high performance trading platform. All of the data generated by SFTI had to be funneled back in to NYSE Euronext's data delivery platform.

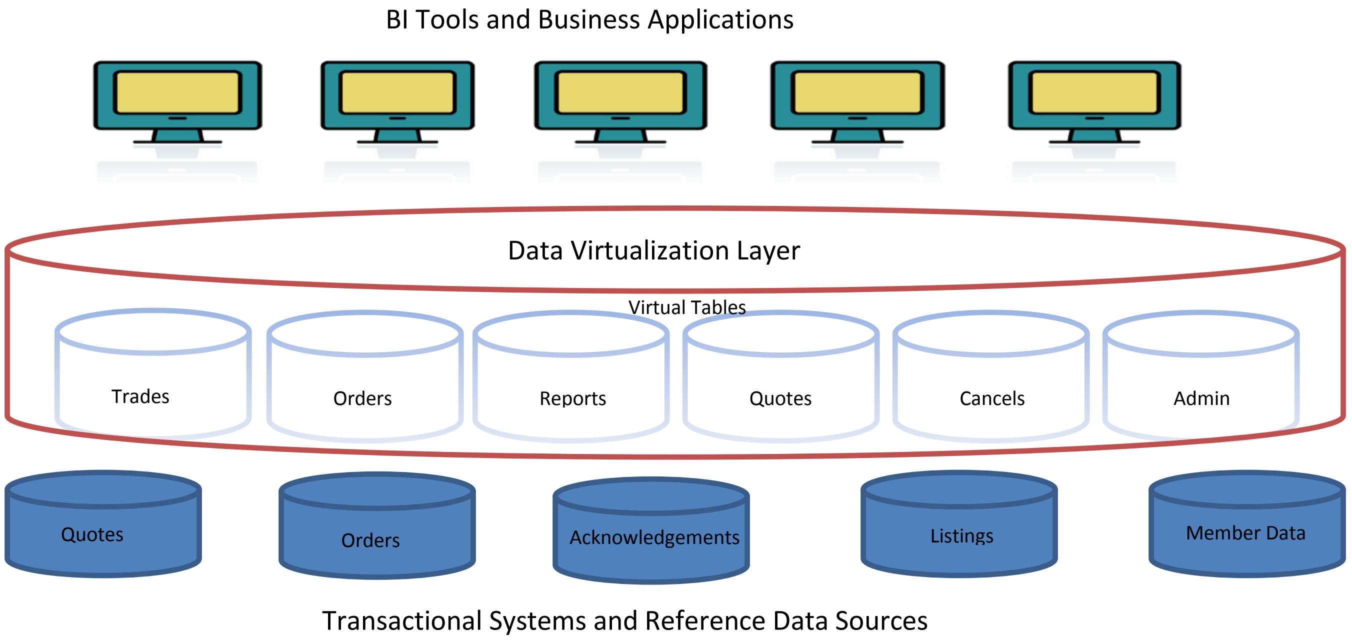
8.1.3 Data Virtualization Solution

In order to succeed in such a complex environment, NYSE Euronext focused on "Automation", "Standardization" and "Performance". NYSE Euronext standardized access to data across the organization by standardizing the architecture and tool set for data access. Prior to deployment of DV, each application needed to install and maintain data access drivers for different databases and also had to manage security and user accounts for each of these databases. Different application would need to bring in data from different data bases which made the management and maintenance difficult. With DV all applications needed only one driver, the driver to talk to DV; all of the authentication and authorization is need only for DV user account. Moreover defining and changing data services is now easy since it only required making changes to SQL defining virtual tables; this made many of the data management changes easy as it didn't require IT organization's help. This allowed business analysts, instead of IT, to maintain logical data layer and to take advantage of their in depth knowledge of domain and data. Standardized data access was used as the foundation for achieving integrated enterprise reference data and streamlined reporting system.

NYSE Euronext used data virtualization to build virtual data warehouse that provided read only access to post trade data. These data comes from different exchanges, which are part of NYSE Euronext, located around the globe. All these exchanges around the globe need to retain data for seven years and need to make data readily accessible. Due to these requirements exchanges deployed multiple DW each storing a different time slice of data. For NYSE Euronext, to see the whole picture of business they needed to integrate data from all the different DW and Data Marts from all the different exchanges around the globe.

Reconciling semantic differences between exchanges on various business entities is a major challenge for the data integration effort at NYSE Euronext. For example there are 300 different ways to describe a "member firm". NYSE Euronext has adopted a phased approach to these data integration challenges. For first phase NYSE Euronext is focusing on providing a single definition for Customer and Product across all the exchanges around the globe.

As part of DV adoption, NYSE Euronext has configured 80% of their database systems to talk to DV. Applications have been redirected to talk to DV as opposed to talking to data bases directly. Security is very critical to NYSE Euronext. By switching applications to talk to a central DV platform as opposed to point connections to individual DB, NYSE Euronext improved security. Further DV allowed NYSE Euronext to track data access patterns; who ran which reports, how long did each report run.


BI Tools and Business Applications
Data Virtualization Layer
Virtual Tables
Trades
Orders
Reports
Quotes
Cancels
Admin
Quotes
Orders
Acknowledgements
Listings
Member Data
Transactional Systems and Reference Data Sources

NYSE Euronext – DV Solution

Figure: 25

Above diagram depicts the high level reference architecture of DV deployment with in NYSE Euronext. Data sources include both transactional systems and referential data systems. "Quotes", "Orders", "Acknowledgements" are examples of transactional data. "Listings", "member data", "customer", "product" are some examples of referential data.

In NYSE Euronext data virtualization layer performed many functions and roles. Some of them are listed below:

- Standardized data access for connectivity to all data sources
- Virtual DW for federating data through logical views
- Centralized repository for common metadata, application/business logic, connectivity and data services

Key benefits of the system are:

- Changes to any of the centralized components are done in the data virtualization layer and thus eliminates need to deploy code changes to applications
- Data virtualization provides extensibility and reusability by virtue of embedding views with in views.
- DV allows for horizontal and vertical federation. Horizontal federation brings data from dissimilar data sources whereas vertical federation slices data based on time.
- Reduced the foot print of software deployed
- Enhanced agility due to flexibility offered by logical views.

8.2 Qualcomm

8.2.1 Organization Background

Qualcomm, a global leader in mobile technology, manufactures chipsets, licenses technology and provides communication services worldwide to telecommunication industry. Qualcomm employs more than 17000 employees and has annual revenue in excess of $10 billion.

8.2.2 Business Problem

IT department in Qualcomm was constantly getting challenged to get things done faster in order for the company to maintain its leadership position in mobile technology market. Agility and speed is especially important for Qualcomm because of the speed at which technology changes in mobile space. The traditional data integration approaches would move data around, needs process in place to do data hand offs and hence would take months to implement.

Another problem was the consistency and integrity of the data. Qualcomm had terabytes of data which got moved in to many different data warehouses and data marts. It was estimated that in some cases data was being copied 10 times and stored in 10 different systems. Monitoring all these systems and maintaining data consistency was costly. Lot of disk space was used to store all these data and considerable man power was employed to maintain all these resources. Also when data got out of sync in all these different copies, considerable time had to be spend to restore data integrity and consistency.

8.2.3 Data Virtualization Solution

The primary motivation for data virtualization was the possibility to make data available to applications without having to copy data around. Prior to DV, data was getting copied in to DW and Data Marts even though many BI applications didn't require historical data. Since DW and Data Marts were the data integration choice prior to DV, building new BI applications often required expending considerable time to implement DW and Data Mart if existing DW and Data Mart couldn't satisfy the requirements. With DV new applications can be built really fast using virtual views of data virtualization as opposed to months spend on doing data integration projects using traditional technology. Qualcomm's enterprise architecture group built a company wide data virtualization layer which are then used by applications.

"People" was the first project that took advantage of data virtualization technology. This was an application to look up people in Qualcomm and gather all their attributes, like their telephone number, physical location and reporting hierarchy. The existing solution was a legacy application written in Java that brought data from LDAP, HR data bases and other systems. This application was in existence for years at Qualcomm and was one of the most heavily used web application. There were primarily two issues with this legacy application:

- Maintaining the legacy application required lot of effort.
- The code to access the legacy app was not reusable; when some other application required the same data, those applications had to redo the effort of “people” app. There was no API to expose data out from “people” application.

Data Virtualization provided an easy way to integrate data from all the systems using virtual views. These virtual views provided extensibility and reusability. With data virtualization, the logic for data integration got pushed down from application to DV platform. Another advantage of DV solution was agility; adding a new data source in old solution required application code changes, with DV adding new data source is easy and inexpensive.

“Oasis” was another key project where DV was used. “Oasis” was a project to enable project managers by providing them more information about chipsets, like overall design, development and manufacturing process, information about cores etc. The data needed by project managers were spread over 10 to 15 systems. Moreover it was expected that 15 more sources of data will be added shortly. The traditional data integration approach would have taken years to implement and it was expected that project won’t complete until 2016. With DV, “Oasis” was completed and operationalized in 18 months.

According to Qualcomm IT departments, the key benefits of DV are:

- Agility and Time-To-Market
- Reduced cost
- More efficient data management
- Better ROI

Chapter 9

Pitfalls of Data Virtualization & Enterprise Data Standardization

9.1 Data Virtualization, Replacement for Data Ware Housing?

Data virtualization even though offers lower implementation time and lower operational cost may not be the best solution for all analytical and reporting use cases. Even though data virtualization can off load many of analytical use cases from DW, Data Warehouses still have significant roles to play in analytical landscape.

Most DW implementations would store historical data for some period of time. There would be policies in place to manage data aging and lifecycle. Historical data is an important part of regression analysis. Many data mining projects may benefit from historical data. Unlike DW, DV doesn't handle data storage except for short term data caches. These short term data caches are typically stored in regular OLTP data bases or flat files. Storing large amounts of historical data requires specialized DB software often assisted by specialized hardware solutions to speed up data access. Teradata, Netezza and Oracle ExData data bases are some examples of the specialized solutions for DW. DV could theoretically use DW for historical data storage; in such a scenario DV primarily acts as a replacement of ETL in the traditional BI approach and doesn't deliver all of the cost and Time-To-Market savings of pure DV solution. So pure DV cannot be a solution for cases where data needs to be retained for long periods of time; in such cases DV can be used to replace ETL, but data itself needs to be stored in some repository like DW.

DV technology is not well suited to capture every change in production data. This is because DV basically has a data access mechanism based on query; the assumption is that data would be fetched only when it is needed by end user and data would be cleansed, transformed and correlated in flight before giving it to the end user thus avoiding intermediate data storage. In other words DV performs data integration on demand. In cases where user wants to capture every change in the production data, DW with some form of Change Data Capture (CDC) mechanisms or Data replication mechanisms may be a better fit.

Most DW implementations would have large amount of data stored in them. Often analytical work load would require processing terabytes of data. DV would be a poor solution if terabytes of data need to be brought in to DV platform for data integration and transformation. The query latency would increase drastically. Since DW handles data storage itself, DW could have indexes built on data. Many times DW employs columnar data storage which helps with data aggregation. Also DW often offloads work load to custom built hardware thus speeding up data processing. When large quantity of data needs to be ingested DW may be a better solution compared to DV. As described above DV could be used to populate DW; such architecture doesn't get all the cost and Time-To-Market savings of pure DV solution and in such hybrid solutions, DV is used rather as a replacement for ETL. Such hybrid architecture would incur the cost of maintaining DW and possibly data marts; this solution would suffer from the inflexibilities involved in changing schemas of DW and data mart which would affect BI project's Time-To-Market.

Many times analysts need to look at data from different dimensions like looking at sales data for a specific quarter from a specific region for a specific product. Most DW offers multi-dimensional data structures like Cubes to help with such analytical workloads. DW often performs data pre aggregation along different dimensions to speed up query performance. For DV it would be hard to match the functionality and performance of DW when multi-dimensional view of data is needed.

Following table captures the main differences between DV and DW solution:

	DV	**DW**
Storage of Historical Data	NO	YES
Capture Every Change in Production Data	NO	YES (requires integration with CDC)
Multi-Dimensional Data Structures	NO	YES
Data Pre-Aggregation	NO	YES
Query performance on large amounts of data	SLOW (relative to DW)	FAST (relative to DV)
Data Integration on Demand	YES	NO
Operational Cost	LOW (relative to DW)	HIGH (relative to DV)
Time-To-Market	LOW (relative to DW)	HIGH (relative to DV)
Easy to Make Changes	YES (relative to DW)	NO (relative to DV)
Dependence on IT	LOW (relative to DW)	HIGH (relative to DV)

9.2 Pitfalls of Enterprise Data Standardization

Enterprise data standardization offers cost savings from fewer data type mismatches and semantic incompatibilities. However Enterprise Data standardization requires discipline and careful evaluation. Standardizing on few vendors for data sources may cause vendor lock in and could inflate software cost in long term.

Maintaining correct semantic definition of business entities would require enforcement of discipline throughout worldwide IT organization. Enforcing discipline across all IT projects all across the globe over period of years may prove to be hard. Another problem is extensibility of the semantic definition. If a new project required adding more elements to existing semantic definition, it may require all of the current applications to ingest those changes as well. Every change to semantic definition could thus become global in scope affecting all applications of the company.

Chapter 10

Conclusions

Success of organizations depends to a great extent on the quality of their decision making. Data driven decision making has a significant role to play in organizational decision making. Organizations are investing more and more and more in technologies to maintain and analyze data. Data integration is a key part of data analysis.

Traditional data integration approaches revolved around physically copying data from production systems to many intermediate systems before it is analyzed. Typical data integration implementation would copy data from production systems to staging area, DW, Data Mart and PDS. This approach is costly, time consuming and inflexible. One of the key issues of data integration is reconciling data type mismatches among data from different data sources and reconciling mismatches in semantics of business entities.

Data Virtualization and Enterprise Data Standardization has the promise of reducing cost and implementation time of data integration. Unlike DW, DV defines data cleansing, data joins and transformations programmatically using logical views. This avoids the cost and inflexibility of data copying involved with traditional approach. DV allows for extensibility and reuse by allowing for chaining of logical view. Enterprise data standardization mostly avoids data type mismatches and semantic incompatibilities in data, thus reducing project implementation time and cost.

DV is not a replacement for DW rather DV could offload certain analytical workloads from DW. Regression analysis, multi-dimensional data structures, analysis of large amounts of data would all mostly require DW. Even though enterprise data standardization could reduce data integration time, it needs to be done after careful evaluation; vendor lock in and inflexibility can be potential pitfalls of data standardization.

Many organizations have already adopted DV and are reaping benefits from it. Some of these use cases are depicted in "Data Virtualization" book, printed Sep 2011, by Judith R. Davis and Robert Eve. Data Virtualization software companies like Composite Software, Denodo, Informatica and IBM seems to be growing their Data Virtualization business. Shorter project implementation times and lower operational cost has been the drivers for DV adoption. Data Virtualization and Enterprise Data standardization is enabling data driven decision making.

References

1. Data Virtualization for Business Intelligence Systems, Rick F. van der Lans, 2012
2. Data Virtualization, Judith R. Davis and Robert Eve, September 2011
3. Forrester Report, Data Virtualization Reaches Critical Mass, Brian Hopkins for Enterprise Architecture Professionals, June 2011
4. Data Cleaning: Problems and Current Approaches by Erhard Rahm, Hong Hai Do, University of Leipzig, Germany, http://dbs.uni-leipzig.de, Dec 2000
5. "Customer Churn Analysis in the Wireless Industry: A Data Mining Approach" by Shyam V. Nath & Dr. Ravi S. Behara, Florida Atlantic University, 2003
6. 90 Days to the Data Mart by Alan Simon 1998
7. Denormalization Effects on The Performance of RDBMS, G. Lawrence Sanders, Seungkyoon Shin, State University of New York, Buffalo, 2002
8. http://en.wikipedia.org/wiki/Denormalization, May 2013
9. Z. Wei, J. Dejun, G. Pierre, C.-H. Chi and M. van Steen. Service-Oriented Data Denormalization for Scalable Web Applications. In Proceedings of the International World-Wide Web conference, April 2008
10. Star Schema: The Complete Reference by Christopher Adamson, July 2010
11. http://en.wikipedia.org/wiki/Snowflake_schema, May 2013
12. Anahory, S.; D. Murray. *Data Warehousing in the Real World: A Practical Guide for Building Decision Support Systems*. Addison Wesley Professional. , July 1997
13. Kimball, Ralph (1996). *The Data Warehousing Toolkit*. John Wiley
14. Why is the Snowflake Schema a Good Data Warehouse Design? By Mark Levene and George Loizou, School of Computer Science and Information Systems, Birkbeck College, University of London, 2003
15. http://en.wikipedia.org/wiki/Star_schema, May 2013
16. Ralph Kimball and Margy Ross, *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling (Second Edition)*, April 2002
17. http://www.dwhworld.com/dwh-schemas/, May 2013
18. "McLeod and Heimbigner (1985). "A Federated Architecture for information management". *ACM Transactions on Information Systems, Volume 3, Issue 3*. pp. 253–278
19. http://en.wikipedia.org/wiki/Data_analysis, May 2013
20. http://en.wikipedia.org/wiki/Data_visualization, May 2013
21. http://www.systemdynamics.org, May 2013